MELFA

Industrieroboter

Bedienungs- und
Programmieranleitung

Steuergeräte
CRnD/CRnQ

23 02 2012
Version C

Bedienung- und Programmieranleitung
Steuergeräte CRnD/CRnQ

Version			Änderungen / Ergänzungen / Korrekturen	
A	09/2010	pdp – gb	—	
B	01/2011	pdp – gb	Allgemein: Abschn. 5.2: Abschn. 6.3.3: Abschn. 7.2.14: Abschn. 7.2.61: Abschn. 9.2: Tab. 10-2:	Diverse Korrekturen und Ergänzungen Abschnitt „Koordinatensysteme des Roboters“ hinzugefügt Modifikation des Base-Befehls Neue Variable „M_BsNo“ hinzugefügt Neue Variable „P_WkCord“ hinzugefügt Neue Parameter „MAPMODE“ und „LOADMODE“ hinzugefügt Neue Parameter „IQMEM“ und „IQSPEC“ hinzugefügt
C	02/2012	pdp – gb	Allgemein: Abschn. 3.6: Abschn. 6.3.75: Kap. 7: Abschn. 9.2: Tab. 10-8	Diverse Korrekturen und Ergänzungen Abschnitt „Weltkoordinaten ändern“ hinzugefügt Neuen Befehl „SpdOpt“ hinzugefügt Einstellbereich der Variablen M_ErrLvl und M_Mode angepasst Neue Parameter „MECAR“, „SPDOPT“, „ITBATERR“, „TIMESYNC“ und „MFREPO“ hinzugefügt Neuen Parameter „TMPOUT“ hinzugefügt

Zu diesem Handbuch

Die in diesem Handbuch vorliegenden Texte, Abbildungen, Diagramme und Beispiele dienen ausschließlich der Erläuterung zur Installation, Bedienung und zum Betrieb der in diesem Handbuch beschriebenen Industrieroboter.

Sollten sich Fragen bezüglich Installation und Betrieb der in diesem Handbuch beschriebenen Geräte ergeben, zögern Sie nicht, Ihr zuständiges Verkaufsbüro oder einen Ihrer Vertriebspartner (siehe Umschlagseite) zu kontaktieren.
Aktuelle Informationen sowie Antworten auf häufig gestellte Fragen erhalten Sie über die Internet-Adresse http://www.mitsubishi-automation.de.

Die MITSUBISHI ELECTRIC EUROPE B.V. behält sich vor, jederzeit technische Änderungen dieses Handbuchs ohne besondere Hinweise vorzunehmen.

© 02/2012

Sicherheitshinweise

Zielgruppe

Dieses Handbuch richtet sich ausschließlich an anerkannt ausgebildete Elektrofachkräfte, die mit den Sicherheitsstandards der Automatisierungstechnik vertraut sind. Projektierung, Installation, Inbetriebnahme, Wartung und Prüfung der Roboter nebst Zubehör dürfen nur von einer anerkannt ausgebildeten Elektrofachkraft, die mit den Sicherheitsstandards der Automatisierungstechnik vertraut ist, durchgeführt werden.

Bestimmungsgemäßer Gebrauch

Die Steuergeräte CRnD und CRnQ sind nur für die Einsatzbereiche vorgesehen, die in diesem Handbuch beschrieben sind. Achten Sie auf die Einhaltung aller im Handbuch angegebenen Kenndaten.

Jede andere darüber hinausgehende Verwendung oder Benutzung gilt als nicht bestimmungsgemäß.

Sicherheitsrelevante Vorschriften

Bei der Projektierung, Installation, Inbetriebnahme, Wartung und Prüfung der Geräte müssen die für den spezifischen Einsatzfall gültigen Sicherheits- und Unfallverhütungsvorschriften beachtet werden.

ACHTUNG:

Im Lieferumfang des Roboters ist ein Sicherheitstechnisches Handbuch enthalten. Dieses Handbuch behandelt alle sicherheitsrelevanten Details zu Aufstellung, Inbetriebnahme und Wartung. Vor einer Aufstellung, Inbetriebnahme oder der Durchführung anderer Arbeiten mit dem oder am Roboter ist dieses Handbuch unbedingt durchzuarbeiten. Alle darin aufgeführten Angaben sind zwingend zu beachten!
Sollte dieses Handbuch nicht im Lieferumfang enthalten sein, wenden Sie sich bitte umgehend an Ihren Mitsubishi-Vertriebspartner.

Darüber hinaus müssen folgende Vorschriften (ohne Anspruch auf Vollständigkeit) beachtet werden:

- VDE-Vorschriften
- Brandverhütungsvorschriften
- Unfallverhütungsvorschriften

Erläuterung zu den Gefahrenhinweisen

In diesem Handbuch befinden sich Hinweise, die für den sachgerechten sicheren Umgang mit dem Roboter wichtig sind.

Die einzelnen Hinweise haben folgende Bedeutung:

GEFAHR:

Bedeutet, dass eine Gefahr für das Leben und die Gesundheit des Anwenders, z. B. durch elektrische Spannung, besteht, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

ACHTUNG:

Bedeutet eine Warnung vor möglichen Beschädigungen des Roboters, seiner Peripherie oder anderer Sachwerte, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Symbolik des Handbuchs

Verwendung von Hinweisen

Hinweise auf wichtige Informationen sind besonders gekennzeichnet und werden folgenderweise dargestellt:

HINWEIS | Hinweistext

Verwendung von Beispielen

Beispiele sind besonders gekennzeichnet und werden folgendermaßen dargestellt:

Beispiel ▽ Beispieltext △

Verwendung von Nummerierungen in Abbildungen

Nummerierungen in Abbildungen werden durch weiße Zahlen in schwarzem Kreis dargestellt und in einer anschließenden Tabelle unter der gleichen Zahl erläutert, z. B.:

❶ ❷ ❸ ❹

Verwendung von Handlungsanweisungen

Handlungsanweisungen sind Schrittfolgen bei der Inbetriebnahme, Bedienung, Wartung u. Ä., die genau in der aufgeführten Reihenfolge durchgeführt werden müssen.

Sie werden fortlaufend durchnummeriert (schwarze Zahlen in weißem Kreis):

① Text

② Text

③ Text

Verwendung von Fußnoten in Tabellen

Hinweise in Tabellen werden in Form von Fußnoten unterhalb der Tabelle (hochgestellt) erläutert. An der entsprechenden Stelle in der Tabelle steht ein Fußnotenzeichen (hochgestellt).

Liegen mehrere Fußnoten zu einer Tabelle vor, werden diese unterhalb der Tabelle fortlaufend nummeriert (schwarze Zahlen in weißem Kreis, hochgestellt):

① Text
② Text
③ Text

Inhaltsverzeichnis

1 Einführung

1.1 Grundlegende Sicherheitshinweise 1-1

2 Funktionen

2.1 Steuergerät 2-1
2.1.1 Bedien- und Signalelemente des Steuergerätes 2-1
2.1.2 LED-Anzeige 2-3
2.2 Teaching Box 2-5
2.2.1 Bedienelemente der Teaching Box R32TB 2-5
2.3 Betriebsrechte 2-7
2.4 Bewegungs- und Steuerfunktionen 2-9

3 Bedienung und Programmierung

3.1 Menübaum 3-1
3.2 Eingabe eines Zeichens 3-5
3.3 Auswahl eines Menüpunkts 3-7
3.4 Roboter im JOG-Betrieb bewegen 3-9
3.4.1 JOG-Betriebsarten 3-9
3.4.2 JOG-Geschwindigkeit einstellen 3-11
3.4.3 Gelenk-JOG-Betrieb 3-11
3.4.4 Werkzeug-JOG-Betrieb 3-12
3.4.5 XYZ-JOG-Betrieb 3-12
3.4.6 3-Achsen-XYZ-JOG-Betrieb 3-13
3.4.7 Kreis-JOG-Betrieb 3-13
3.4.8 Werkstück-JOG-Betrieb 3-14
3.4.9 Kollisionsüberwachung im JOG-Betrieb 3-15
3.5 Werkzeugdaten umschalten 3-17
3.6 Weltkoordinaten ändern (Festlegung der Nummer des Basiskoordinatensystems) 3-19

3.7 Handgreifer öffnen/schließen ... 3-21
3.8 Handgreifer ausrichten ... 3-22
3.9 Programmierung ... 3-24
3.9.1 Roboterprogramm erstellen ... 3-24
3.9.2 Roboterprogramm editieren ... 3-28
3.9.3 Roboterprogramm testen ... 3-37
3.10 Automatikbetrieb ... 3-44
3.10.1 Geschwindigkeit einstellen ... 3-44
3.10.2 Auswahl der Programmnummer ... 3-45
3.10.3 Starten des Automatikbetriebs ... 3-46
3.10.4 Stoppen des Automatikbetriebs ... 3-48
3.10.5 Fortsetzung des Automatikbetriebs aus dem Stoppzustand ... 3-48
3.10.6 Programm zurücksetzen ... 3-48
3.11 Servospannung ein-/ausschalten ... 3-50
3.12 Fehler zurücksetzen ... 3-52
3.13 Fehler temporär zurücksetzen ... 3-52
3.14 Programmverwaltungsfunktionen ... 3-53
3.14.1 Programmverzeichnis anzeigen ... 3-53
3.14.2 Programm kopieren ... 3-54
3.14.3 Programmnamen ändern ... 3-55
3.14.4 Programm löschen ... 3-56
3.14.5 Programm schützen ... 3-58
3.14.6 Programm auswählen ... 3-60
3.15 RUN-Menü ... 3-61
3.15.1 Anzeige der Zeile, die gerade ausgeführt wird ... 3-61
3.16 Testbetrieb ... 3-67
3.17 Monitor-Funktionen ... 3-68
3.17.1 Monitor-Funktion für Eingangssignale ... 3-68
3.17.2 Monitor-Funktion für Ausgangssignale ... 3-70
3.17.3 Monitor-Funktion für Eingangsregister ... 3-72
3.17.4 Monitor-Funktion für Ausgangsregister ... 3-74
3.17.5 Monitor-Funktion für Variable ... 3-76
3.17.6 Liste der aufgetretenen Fehlermeldungen ... 3-78
3.18 Parametermenü ... 3-79
3.18.1 Parameter einstellen ... 3-79

3.19 Gelenkbremsen lösen 3-81

3.20 Einstellungs- und Initialisierungsmenü 3-83
3.20.1 Alle gespeicherten Programme löschen 3-83
3.20.2 Alle Parameter auf die Werkseinstellung zurücksetzen 3-85
3.20.3 Batteriezähler zurücksetzen 3-86
3.20.4 Verbleibende Betriebszeit der Batterie und Einschaltzeit anzeigen 3-87
3.20.5 Datum und Uhrzeit einstellen 3-88
3.20.6 Software-Version der Teaching Box und des Steuergeräts anzeigen 3-89

3.21 Zusatzfunktionen 3-90

3.22 Einstellungen im Startmenü 3-91
3.22.1 Auswahl der Landessprache 3-91
3.22.2 Einstellung des Kontrasts 3-94

4 MELFA-BASIC-V-Programmierung

4.1 Funktionsübersicht 4-1

4.2 Programmaufbau 4-2
4.2.1 Programmname 4-2
4.2.2 Anweisung 4-2
4.2.3 Variable 4-3

4.3 Steuerung der Roboterbewegung 4-5
4.3.1 Gelenk-Interpolation 4-5
4.3.2 Linear-Interpolation 4-7
4.3.3 Kreis-Interpolation 4-9
4.3.4 Kontinuierliche Bewegung 4-11
4.3.5 Beschleunigungs-/Bremszeit und Geschwindigkeit 4-13
4.3.6 Feinpositionierung 4-16
4.3.7 Verfahrweggenauigkeit 4-18
4.3.8 Hand- und Werkzeugsteuerung 4-20

4.4 Palettierung 4-22

4.5 Programmsteuerung 4-30
4.5.1 Verzweigungen und Wartezeit 4-30
4.5.2 Programmschleife 4-32
4.5.3 Interrupt 4-33
4.5.4 Unterprogramm 4-34
4.5.5 Timer 4-35
4.5.6 Stopp 4-36

4.6 Ein- und Ausgabe externer Signale 4-37
4.6.1 Eingangssignale 4-37
4.6.2 Ausgangssignale 4-38
4.7 Kommunikation 4-39
4.8 Ausdrücke und Operationen 4-41
4.8.1 Übersicht 4-41
4.8.2 Relative Konvertierung von Positionsdaten 4-43
4.9 Angehängte Anweisung 4-45
4.10 Unterschiede zwischen MELFA-BASIC V und MELFA-BASIC IV 4-46
4.10.1 MELFA-BASIC V 4-46
4.10.2 Merkmale der Programmiersprache MELFA-BASIC V 4-46
4.11 Multitasking-Funktion 4-47
4.11.1 Beschreibung 4-47
4.11.2 Ausführung eines Multitaskings 4-48
4.11.3 Betriebszustand eines Programmplatzes 4-49
4.11.4 Erstellung eines Multitasking-Programms 4-52
4.11.5 Anwendung des Multitaskings 4-54
4.11.6 Beispiel zur Anwendung der Multitasking-Funktion 4-55
4.11.7 Programmspeicherkapazität 4-58

5 MELFA-BASIC V

5.1 Begriffserklärung 5-1
5.1.1 Anweisung 5-1
5.1.2 Angehängte Anweisung 5-1
5.1.3 Programmschritt 5-2
5.1.4 Schrittnummern und Marken 5-2
5.1.5 Zeichentypen 5-3
5.1.6 Zeichen mit besonderer Bedeutung 5-4
5.1.7 Datentypen 5-5
5.1.8 Konstanten 5-6
5.1.9 Variablen 5-11
5.1.10 Externe Variablen 5-15
5.1.11 Funktionen 5-24
5.1.12 Operanden 5-28
5.1.13 Rangfolge von Operationen 5-30
5.1.14 Programmebenen 5-30
5.1.15 Reservierte Wörter 5-30

5.2 Koordinatensysteme des Roboters 5-31
5.2.1 Beschreibung der Koordinatensysteme 5-31
5.2.2 Basis-Konvertierung 5-32
5.2.3 Aufbau der Positionsdaten 5-34
5.2.4 Werkzeugkoordinatensystem (Handflanschkoordinatensystem) 5-35

6 MELFA-BASIC-V-Befehle

6.1 Allgemeine Hinweise 6-1
6.1.1 Beschreibung des verwendeten Formats 6-1
6.2 Übersicht der MELFA-BASIC-V-Befehle 6-2
6.2.1 Alphabetische Übersicht 6-2
6.2.2 Anwendungsspezifische Übersicht 6-5
6.3 Detaillierte Befehlsbeschreibung 6-8
6.3.1 Accel (Accelerate) 6-8
6.3.2 Act (Act) 6-10
6.3.3 Base (Base) 6-12
6.3.4 CallP (Call P) 6-16
6.3.5 ChrSrch (Character search) 6-18
6.3.6 Close (Close) 6-19
6.3.7 Clr (Clear) 6-20
6.3.8 Cmp Jnt (Compliance Joint) 6-21
6.3.9 Cmp Pos (Compliance Posture) 6-23
6.3.10 Cmp Tool (Compliance Tool) 6-26
6.3.11 Cmp Off (Compliance OFF) 6-29
6.3.12 CmpG (Compliance Gain) 6-30
6.3.13 Cnt (Continuous) 6-32
6.3.14 ColChk (Col Check) 6-36
6.3.15 ColLvl (Col Level) 6-40
6.3.16 Com Off (Communication OFF) 6-42
6.3.17 Com On (Communication ON) 6-43
6.3.18 Com Stop (Communication STOP) 6-44
6.3.19 Def Act (Define act) 6-45
6.3.20 Def Arch (Define Arch) 6-48
6.3.21 Def Char (Define Character) 6-50
6.3.22 Def Fn (Define function) 6-51
6.3.23 Def Inte/Long/Float/Double (Define Integer/Long/Float/Double) 6-53
6.3.24 Def IO (Define IO) 6-55

6.3.25 Def Jnt (Define Joint) 6-57
6.3.26 Def Plt (Define pallet) 6-58
6.3.27 Def Pos (Define Position) 6-61
6.3.28 Dim (Dim) 6-62
6.3.29 Dly (Delay) 6-64
6.3.30 End (End) 6-66
6.3.31 Error (Error) 6-67
6.3.32 Fine (Fine) 6-68
6.3.33 Fine J (Fine Joint) 6-70
6.3.34 Fine P (Fine Pause) 6-72
6.3.35 For-Next (For-Next) 6-74
6.3.36 FPrm (FPRM) 6-76
6.3.37 GetM (Get Mechanism) 6-77
6.3.38 GoSub (Go Subroutine) 6-79
6.3.39 GoTo (Go To) 6-80
6.3.40 Hlt (Halt) 6-81
6.3.41 HOpen/HClose (Hand Open/Hand Close) 6-82
6.3.42 If ... Then ... Else (If Then Else) 6-83
6.3.43 Input # (Input) 6-85
6.3.44 JOvrd (J Override) 6-86
6.3.45 JRC (Joint Roll Change) 6-87
6.3.46 Label (Label) 6-90
6.3.47 LoadSet (Load Set) 6-91
6.3.48 Mov (Move) 6-93
6.3.49 Mva (Move Arch) 6-95
6.3.50 Mvc (Move C) 6-97
6.3.51 Mvr (Move R) 6-99
6.3.52 Mvr2 (Move R2) 6-101
6.3.53 Mvr3 (Move R3) 6-103
6.3.54 Mvs (Move S) 6-105
6.3.55 MvTune (Move Tune) 6-108
6.3.56 Mxt (Move External) 6-111
6.3.57 Oadl (Optimum Acceleration/Deceleration) 6-113
6.3.58 On Com GoSub (ON Communication Go Subroutine) 6-115
6.3.59 On GoSub (ON Go Subroutine) 6-117
6.3.60 On GoTo (ON Go To) 6-119
6.3.61 Open (Open) 6-121
6.3.62 Ovrd (Override) 6-123
6.3.63 Plt (Pallet) 6-125

6.3.64 Prec (Precision) 6-128
6.3.65 Print # (Print) 6-129
6.3.66 Priority (Priority) 6-131
6.3.67 RelM (Release Mechanism) 6-132
6.3.68 Rem (Remarks) 6-133
6.3.69 Reset Err (Reset Error) 6-134
6.3.70 Return (Return) 6-135
6.3.71 Select Case (Select Case) 6-137
6.3.72 Servo (Servo) 6-139
6.3.73 Skip (Skip) 6-140
6.3.74 Spd (Speed) 6-141
6.3.75 SpdOpt (Speed Optimize) 6-143
6.3.76 Title (Title) 6-146
6.3.77 Tool (Tool) 6-147
6.3.78 Torq (Torque) 6-149
6.3.79 Wait (Wait) 6-151
6.3.80 While ~ WEnd (While End) 6-152
6.3.81 Wth (With) 6-153
6.3.82 WthIf (With If) 6-154
6.3.83 XClr (X Clear) 6-155
6.3.84 XLoad (X Load) 6-156
6.3.85 XRst (X Reset) 6-157
6.3.86 XRun (X Run) 6-158
6.3.87 XStp (X Stop) 6-160
6.3.88 Substitute (Substitute) 6-161

7 Roboterstatusvariablen

7.1 Allgemeine Hinweise 7-1
7.1.1 Beschreibung des verwendeten Formats 7-1

7.2 Detaillierte Variablenbeschreibung 7-2
7.2.1 C_Com 7-2
7.2.2 C_Date 7-4
7.2.3 C_Maker 7-4
7.2.4 C_Mecha 7-5
7.2.5 C_Prg 7-5
7.2.6 C_Time 7-6
7.2.7 C_User 7-6
7.2.8 J_ColMxl 7-7

7.2.9 J_Curr 7-9
7.2.10 J_ECurr 7-9
7.2.11 J_Fbc/J_AmpFbc 7-10
7.2.12 J_Origin 7-11
7.2.13 M_Acl/M_DAcl/M_NAcl/M_NDAcl/M_AclSts 7-12
7.2.14 M_BsNo 7-13
7.2.15 M_BrkCq 7-14
7.2.16 M_BTime 7-15
7.2.17 M_CmpDst 7-16
7.2.18 M_CmpLmt 7-17
7.2.19 M_ColSts 7-18
7.2.20 M_Cstp 7-19
7.2.21 M_Cys 7-19
7.2.22 M_DIn/M_DOut 7-20
7.2.23 M_Err/M_ErrLvl/M_Errno 7-21
7.2.24 M_Exp 7-23
7.2.25 M_Fbd 7-24
7.2.26 M_G 7-25
7.2.27 M_HndCq 7-25
7.2.28 M_In/M_Inb/M_In8/M_Inw/M_In16 7-26
7.2.29 M_In32 7-28
7.2.30 M_JOvrd/M_NJOvrd/M_OPOvrd/M_Ovrd/M_NOvrd 7-30
7.2.31 M_LdFact 7-31
7.2.32 M_Line 7-33
7.2.33 M_Mode 7-34
7.2.34 M_On/M_Off 7-34
7.2.35 M_Open 7-35
7.2.36 M_Out/M_Outb/M_Out8/M_Outw/M_Out16 7-37
7.2.37 M_Out32 7-40
7.2.38 M_PI 7-44
7.2.39 M_Psa 7-44
7.2.40 M_Ratio 7-45
7.2.41 M_RDst 7-46
7.2.42 M_Run 7-47
7.2.43 M_SetAdl 7-48
7.2.44 M_SkipCq 7-50
7.2.45 M_Spd/M_NSpd/M_RSpd 7-51
7.2.46 M_Svo 7-52
7.2.47 M_Timer 7-53

7.2.48 M_Tool 7-54
7.2.49 M_Uar 7-56
7.2.50 M_Uar32 7-57
7.2.51 M_Wai 7-59
7.2.52 M_Wupov 7-60
7.2.53 M_Wuprt 7-61
7.2.54 M_Wupst 7-62
7.2.55 P_Base/P_NBAase 7-63
7.2.56 P_ColDir 7-64
7.2.57 P_Curr 7-66
7.2.58 P_Fbc 7-67
7.2.59 P_Safe 7-68
7.2.60 P_Tool/P_NTool 7-69
7.2.61 P_WkCord 7-70
7.2.62 P_Zero 7-72

8 Funktionen

8.1 Allgemeine Hinweise 8-1
8.1.1 Beschreibung des verwendeten Formats 8-1

8.2 Detaillierte Funktionsbeschreibung 8-2
8.2.1 Abs 8-2
8.2.2 Align 8-3
8.2.3 Asc 8-4
8.2.4 Atn/Atn2 8-5
8.2.5 Bin$ 8-6
8.2.6 CalArc 8-7
8.2.7 Chr$ 8-9
8.2.8 Cint 8-9
8.2.9 CkSum 8-10
8.2.10 Cos 8-11
8.2.11 Cvi 8-11
8.2.12 Cvs 8-12
8.2.13 Cvd 8-12
8.2.14 Deg 8-13
8.2.15 Dist 8-14
8.2.16 Exp 8-14
8.2.17 Fix 8-15
8.2.18 Fram 8-16

8.2.19 Hex$ 8-17
8.2.20 Int. 8-18
8.2.21 Inv 8-18
8.2.22 JtoP 8-19
8.2.23 Left$ 8-20
8.2.24 Len. 8-20
8.2.25 Ln 8-21
8.2.26 Log 8-21
8.2.27 Max 8-22
8.2.28 Mid$ 8-23
8.2.29 Min 8-24
8.2.30 Mirror$ 8-24
8.2.31 Mki$ 8-25
8.2.32 Mks$ 8-26
8.2.33 Mkd$ 8-27
8.2.34 PosCq 8-27
8.2.35 PosMid 8-28
8.2.36 PtoJ 8-29
8.2.37 Rad 8-30
8.2.38 Rdfl1 8-31
8.2.39 Rdfl2 8-32
8.2.40 Rnd 8-33
8.2.41 Right$ 8-34
8.2.42 Setfl1 8-35
8.2.43 Setfl2 8-37
8.2.44 SetJnt 8-38
8.2.45 SetPos. 8-39
8.2.46 Sgn 8-40
8.2.47 Sin 8-40
8.2.48 Sqr 8-41
8.2.49 Strpos 8-41
8.2.50 Str$ 8-42
8.2.51 Tan. 8-42
8.2.52 Val 8-43
8.2.53 Zone 8-44
8.2.54 Zone2 8-46
8.2.55 Zone3 8-47

9 Parameter

9.1 Allgemeines 9-1

9.2 Bewegungsparameter 9-2

9.3 Signalparameter 9-17
9.3.1 Eingabe-Offset bei Multi-CPUs (nur für die Steuerung CRnQ) 9-22

9.4 Betriebsparameter 9-24

9.5 Befehlsparameter 9-27

9.6 Kommunikationsparameter 9-31

9.7 Standard-Werkzeugkoordinaten 9-34
9.7.1 Aufbau der Werkzeugdaten 9-34

9.8 Standard-Basiskoordinaten 9-37
9.8.1 Aufbau der Basiskoordinatendaten 9-37

9.9 Benutzerdefinierter Bereich 9-39
9.9.1 Auswahl des Koordinatensystems 9-39
9.9.2 Festlegung der Bereiche 9-41
9.9.3 Mechanismusauswahl für den Begrenzungsbereich 9-43
9.9.4 Festlegung der Bereichsprüfmethode 9-43

9.10 Verfahrwegbegrenzungsebene 9-45

9.11 Automatische Rückkehr 9-46

9.12 Automatischer Programmstart nach dem Einschalten 9-48

9.13 Handgreifer 9-50

9.14 Handgreiferzustand nach Initialisierung 9-51

9.15 Ausgangsbitmuster 9-53

9.16 Kommunikationseinstellungen (RS232) 9-56
9.16.1 Allgemeine Beschreibung 9-56

9.17 Kommunikationseinstellungen (Ethernet) 9-57
9.17.1 Vorgaben der CE-Norm für die Anschlussleitung 9-57
9.17.2 Parametereinstellungen 9-58
9.17.3 Beispieleinstellung für Parameter 1 9-61
9.17.4 Beispieleinstellung für Parameter 2-1 9-62
9.17.5 Beispieleinstellung für Parameter 2-2 9-63
9.17.6 Beispieleinstellung für Parameter 3 9-65

9.18 Überprüfen der Verbindung 9-66
9.18.1 Prüfung der Verbindung mit dem ping-Befehl von Windows 9-66

9.19 Hand- und Werkstückbedingung 9-67
9.19.1 Optimale Beschleunigung/Abbremsung 9-67
9.19.2 Handgreiferzustand 9-68
9.19.3 Definition der Koordinatensysteme für die Hand- und Werkstückbedingungen 9-69
9.20 Fehlermeldung bei Erreichen des singulären Punktes 9-70
9.21 ROM- und Highspeed-RAM-Modus 9-71
9.21.1 Übersicht 9-71
9.21.2 Umschaltung zwischen ROM- und RAM-Modus 9-76
9.21.3 Umschaltung in den ROM-Modus 9-77
9.21.4 Anzeigen im ROM-Modus 9-79
9.21.5 Programmeditierung im ROM-Modus 9-79
9.21.6 Umschaltung in den RAM-Modus 9-80
9.21.7 Umschaltung in den Highspeed-RAM-Modus 9-82
9.22 Warmlaufbetrieb 9-83
9.22.1 Funktionsbeschreibung 9-83
9.22.2 Aktivierung des Warmlaufbetriebs 9-83
9.22.3 Aktivierter Warmlaufbetrieb 9-84
9.22.4 Parameter, spezielle Ein- und Ausgänge und Statusvariablen im Warmlaufbetrieb 9-86
9.22.5 Ausführung des Warmlaufbetriebs 9-87
9.22.6 Wenn der Warmlaufbetrieb freigegeben ist 9-88
9.22.7 Umschaltung zwischen Normal- und Warmlaufbetrieb 9-89
9.22.8 Alarme im Warmlaufbetrieb 9-91
9.23 Durchfahren eines singulären Punktes 9-92
9.23.1 Positionen singulärer Punkte, die durchfahren werden können 9-92
9.23.2 Betrieb mit aktivierter Funktion zum Durchfahren singulärer Punkte 9-93
9.23.3 Aktivierung der Funktion zum Durchfahren singulärer Punkte 9-95
9.23.4 Funktion zum Durchfahren singulärer Punkte im JOG-Betrieb 9-95
9.23.5 Funktion zum Durchfahren singulärer Punkte beim Anfahren definierter Positionen 9-96
9.23.6 Funktion zum Durchfahren singulärer Punkte im Automatikbetrieb 9-96
9.23.7 TYPE (Type) 9-96
9.23.8 Kollisionsüberwachung 9-98
9.23.9 Funktionsbeschreibung 9-98
9.23.10 Robotermodelle mit Kollisionsüberwachung 9-99
9.23.11 Parameter in Zusammenhang mit der Kollisionsüberwachung 9-100
9.23.12 Einsatz der Kollisionsüberwachung 9-101

10 Externe Ein-/Ausgänge

10.1 Einteilung 10-1
10.1.1 Allgemeine Übersicht der Ein- und Ausgänge 10-1
10.2 E/A-Link-Funktion für die SPS 10-2
10.2.1 Parametereinstellung 10-2
10.2.2 Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale 10-5
10.2.3 SPS-Programm (Kontaktplan) 10-6
10.2.4 Zuordnung der speziellen Ein-/Ausgänge 10-8
10.2.5 E/A-Adressanzahl der Modellreihen CRnQ-700 und CRn-500 10-10
10.3 Belegung der speziellen Ein- und Ausgänge 10-11
10.3.1 Programmsteuerung durch externe Signale 10-35
10.4 NOT-HALT-Eingang 10-41
10.4.1 Verhalten des Roboters bei Betätigung des NOT-HALT-Schalters 10-43
10.5 Anschluss eines Bediengerätes der Serie GOT1000 10-44
10.5.1 Konfigurationsbeispiel 10-44
10.5.2 Technische Daten des Bediengeräts 10-45
10.5.3 Ethernet-Anschluss an das Steuergerät 10-46
10.5.4 Einstellungen 10-47

11 Echtzeitsteuerung & Programmbeispiele

11.1 Externe Steuerung in Echtzeit 11-1
11.1.1 Erläuterung der Kommunikation mit Datenpaketen 11-3
11.2 Beispielprogramme 11-6
11.2.1 Beispielprogramm der Datenverbindung (Daten-Link) 11-6
11.2.2 Programm für einen Client 11-10
11.2.3 Programm für einen Server 11-11
11.2.4 Beispielprogramm für externe Echtzeitsteuerung 11-13
11.3 Stellungsmerker 11-23
11.3.1 Beim Knickarmroboter 11-23
11.3.2 Beim SCARA-Roboter 11-25

A Anhang

A.1 Fehlerdiagnose A-1
A.1.1 Übersicht der Fehlercodes A-2

1 Einführung

Die in diesem Handbuch vorliegenden Texte, Abbildungen, Diagramme und Beispiele gelten für folgende Software-Versionen:

- Steuergeräte
  CRnD ab Software-Version S2g
  CRnQ ab Software-Version R2g
- Teaching Box R32TB ab Software-Version 1.6

1.1 Grundlegende Sicherheitshinweise

Der MELFA-Roboter ist nach dem neuesten Stand der Technik gebaut und betriebssicher ausgeführt. Ungeachtet dessen können von dem Roboter Gefahren ausgehen, wenn er nicht von geschultem oder zumindest eingewiesenem Personal betrieben wird oder unsachgemäß bzw. zu nicht bestimmungsgemäßem Gebrauch eingesetzt wird.

Dies betrifft insbesondere:

- **Gefahren für Leib und Leben des Benutzers oder Dritter**
- **Beeinträchtigungen des Roboters, anderer Maschinen und weiterer Sachwerte des Anwenders**

ACHTUNG:

Jede Person, die im Betrieb des Anwenders mit der Aufstellung, Inbetriebnahme, Bedienung, Wartung und Reparatur des Roboters beauftragt ist, muss neben der zum Roboter gehörenden Technischen Dokumentation besonders das mitgelieferte

SICHERHEITSTECHNISCHE HANDBUCH

gelesen und verstanden haben.

ACHTUNG:

Achten Sie strikt auf die Einhaltung aller Sicherheitsrichtlinien. Im Rahmen dieser einführenden Sicherheitshinweise werden folgende weitere Instruktionen gegeben:

Der Roboter darf nur von ausgebildetem und autorisiertem Bedienungspersonal betrieben und bedient werden.

Die Zuständigkeiten für die unterschiedlichen Tätigkeiten im Rahmen des Betreibens des Roboters müssen klar festgelegt und eingehalten werden, damit unter dem Aspekt der Sicherheit keine unklaren Kompetenzen auftreten.

Bei allen Arbeiten, die die Aufstellung, die Inbetriebnahme, das Rüsten, den Betrieb, Änderungen der Einsatzbedingungen und Betriebsweisen, Wartung, Inspektion und Reparatur betreffen, sind die in der Betriebsanleitung angegebenen Ausschaltprozeduren zu beachten.

Die Lage der NOT-HALT-Taster muss bekannt sein und die NOT-HALT-Taster müssen jederzeit zugänglich sein.

Es ist jede Arbeitsweise zu unterlassen, die die Sicherheit an der Maschine beeinträchtigt.

Der Bediener hat dafür zu sorgen, dass keine Personen an dem Roboter arbeiten, die nicht dazu autorisiert sind (z. B. auch durch Betätigung von Einrichtungen gegen unbefugtes Benutzen).

Das verwendende Unternehmen hat dafür zu sorgen, dass der Roboter immer nur in einwandfreiem Zustand betrieben wird.

Der Verwenderbetrieb sollte das zuständige Bedienungspersonal besonders schulen und dazu verpflichten, alle Wartungs- und Inspektionsarbeiten ausschließlich bei abgeschaltetem Roboter und ausgeschalteter Peripherie durchzuführen.

GEFAHR:

Das Steuergerät darf ausschließlich über einen Leistungsschalter an die Netzspannung angeschlossen werden. Bei Nichtbeachtung besteht die Gefahr eines elektrischen Schlages.

Eine detaillierte Beschreibung des Netzanschlusses finden Sie im Technischen Handbuch des Roboters.

2 Funktionen

2.1 Steuergerät

2.1.1 Bedien- und Signalelemente des Steuergerätes

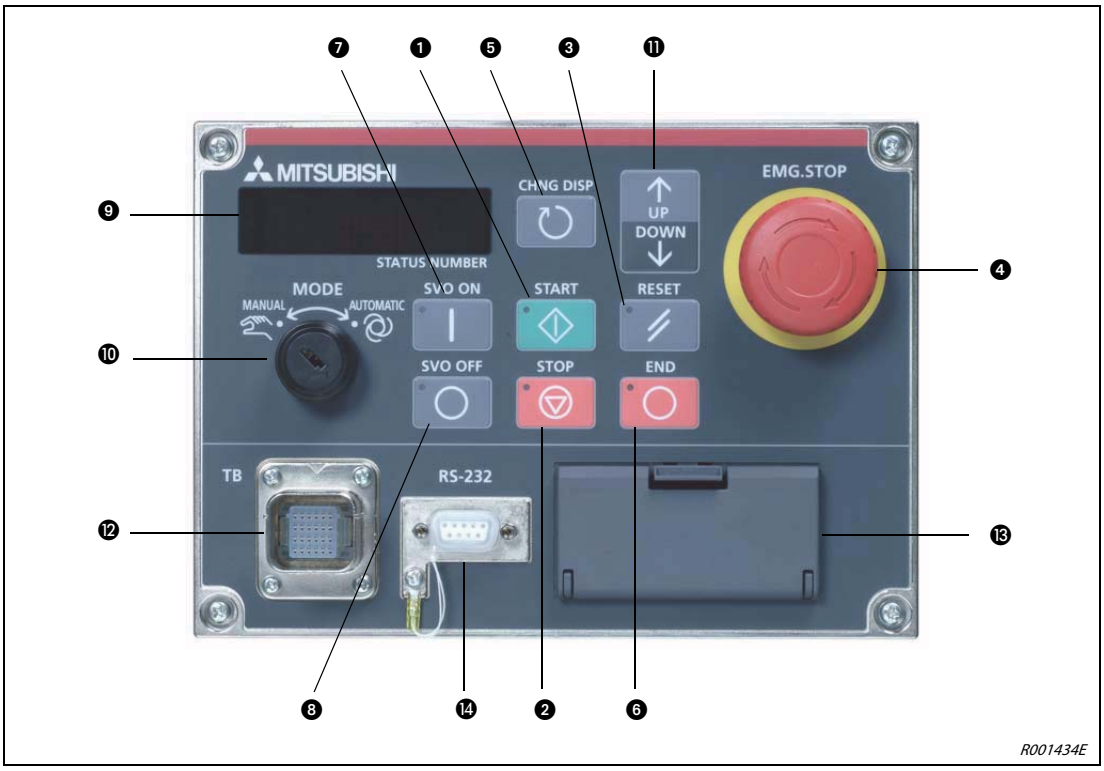


Abb. 2-1: *Vorderansicht des Bedienfeldes*

Nr.	Bezeichnung	Funktion
❶	[START]-Taste	Starten eines Programms und Betrieb des Roboterarms Das Programm wird kontinuierlich abgearbeitet.
❷	[STOP]-Taste	Unterbrechung des laufenden Programms und Abbremsen des Roboters Die Funktion entspricht der Funktion der [STOP]-Taste auf der Teaching Box.
❸	[RESET]-Taste	Quittiert einen Fehlercode Setzt den Haltezustand des Programms und das Programm zurück
❹	[EMG.STOP]-Schalter	Der Rastschalter dient dem NOT-HALT des Robotersystems. Nach Betätigung des Schalters wird die Servospannungsversorgung unmittelbar abgeschaltet und der sich bewegende Roboterarm hält sofort an. Durch Rechtsdrehen wird der Schalter entriegelt und springt wieder heraus.
❺	[CHNG DISP]-Taste	Anzeigewechsel auf dem Display des Steuergeräts in der Reihenfolge: Programmnummer → Zeilennummer → Übersteuerung → Benutzerinformationen → Herstellerinformationen
❻	[END]-Taste	Stoppen des laufenden Programms bei der END-Anweisung
❼	[SVO ON]-Taste	Einschalten der Servoversorgungsspannung
❽	[SVO OFF]-Taste	Abschalten der Servoversorgungsspannung

Tab. 2-1: *Übersicht der Bedien-/Signalelemente des Steuergeräts (1)*

Nr.	Bezeichnung		Funktion
❾	[STATUS NUMBER]-Anzeige		Anzeige von Alarm-, Programmnummer, Übersteuerungswert (%) usw.
❿	[MODE]-Umschalter	AUTOMATIC	Ein Betrieb über das Steuergerät oder externe Geräte ist möglich. Der Betrieb über externe Signale oder die Teaching Box ist deaktiviert. Die Verbindung zwischen Steuergerät und externen Geräten muss durch den Parameter zur Vergabe der Betriebsrechte freigegeben sein. Eine detaillierte Beschreibung der Betriebsrechte finden Sie im Abschn. 2.3.
		MANUAL	Bei aktivierter Teaching Box ist ausschließlich ein Betrieb über die Teaching Box möglich. Der Betrieb über externe Signale oder das Steuergerät ist nicht möglich.
⓫	[UP/DOWN]-Taste		Blättern innerhalb der Anzeige
⓬	T/B-Anschluss		Schnittstelle für den Anschluss der Teaching Box. Stecken Sie den Kurzschlussstecker auf den Schnittstellenanschluss, falls die Teaching Box nicht angeschlossen ist.
⓭	Schnittstelleabdeckung		USB-Schnittstelle und Batterie
⓮	RS232-Schnittstelle (nur CRn-D)		Die RS232-Schnittstelle dient zum Anschluss eines Personalcomputers.

Tab. 2-1: *Übersicht der Bedien-/Signalelemente des Steuergeräts (2)*

HINWEIS Die Taster ❶, ❷, ❸, ❻, ❼ und ❽ haben integrierte Kontrollanzeigen.

2.1.2 LED-Anzeige

Die LED-Anzeige kann über die Tasten [CHNG DISP, [↑ UP] und [↓ DOWN] geändert werden.

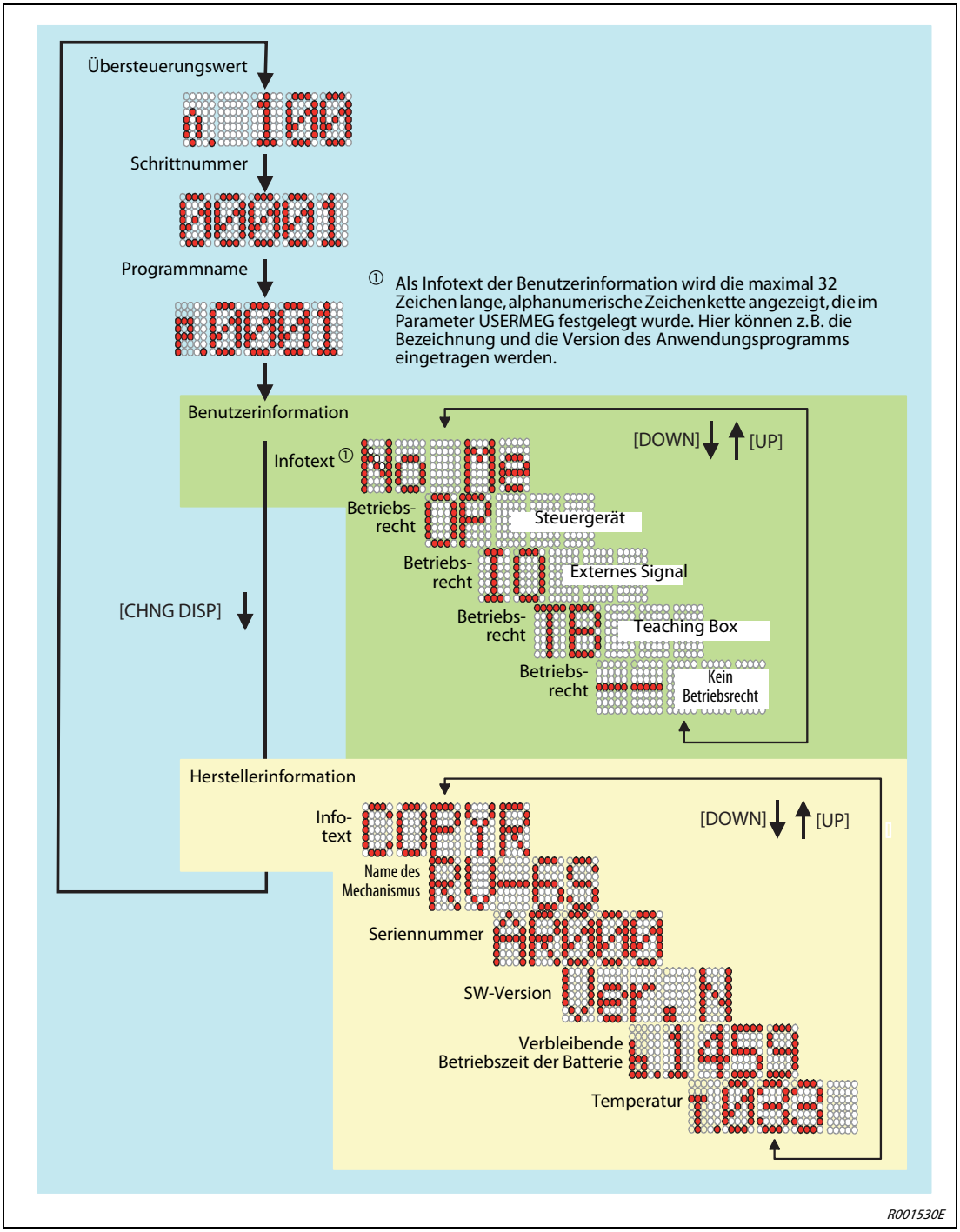
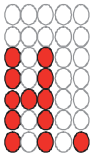
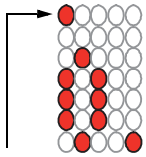


Abb. 2-2: *Status-Number-Anzeige*

Die unterschiedlichen Statusanzeigen werden durch das linke Zeichen näher charakterisiert:

Übersteuerung	Programmname	Verbleibende Betriebszeit der Batterie	Temperatur
Schwerer Fehler	**Leichter Fehler**	**Warnung**	**Betriebsrecht**
			 Punkt leuchtet, wenn das Steuergerät über das Betriebsrecht verfügt.

Tab. 2-2: *Bedeutung der Statusanzeigen*

2.2 Teaching Box

2.2.1 Bedienelemente der Teaching Box R32TB

Abb. 2-3: *Ansichten der Teaching Box R32TB*

Nr.	Bezeichnung	Funktion
❶	[EMG.STOP]-Schalter	NOT-HALT-Schalter mit Verriegelungsfunktion Wenn Sie den Schalter betätigen, hält der Roboterarm sofort an. Die Servoversorgungsspannung wird abgeschaltet. Durch Drehen des Schalters im Uhrzeigersinn wird der Schalter wieder entriegelt.
❷	[ENABLE/DISABLE]-Schalter	Freigabe der Steuerung über die Teaching Box Bringen Sie den Schalter in die Stellung „ENABLE“, um den Roboterarm über die Teaching Box anzusteuern. Wenn die Teaching Box aktiv ist, kann weder über das Bedienfeld des Steuergeräts noch extern in die Steuerung des Roboterarms eingegriffen werden.
❸	Dreistufenschalter	Der dreistufige Zustimmtaster muss für das Einschalten des Servoantriebs bei eingeschalteter Teaching Box betätigt sein.
❹	LCD-Anzeige	Auf der LCD-Anzeige wird der Programm- oder Roboterarmzustand angezeigt.
❺	Statusanzeige	Die LED zeigt den Zustand des Roboters oder Teaching Box an.
❻	[F1]-, [F2]-, [F3]-, [F4]-Taste	Ausführung der aktuell auf dem Display angezeigten Funktionen
❼	[FUNCTION]-Taste	Umschaltung der angezeigten Funktionen
❽	[STOP]-Taste	Unterbrechung des laufenden Programms und Abbremsen des Roboters Die Funktion entspricht der Funktion der [STOP]-Taste auf dem Bedienfeld des Steuergeräts. Die Tastenfunktion ist, unabhängig von der Stellung des [ENABLE/DISABLE]-Schalters, immer verfügbar.
❾	[OVRD ↑]-, [OVRD ↓]-Taste	Änderung der Verfahrgeschwindigkeit Bei Betätigung der [OVRD ↑]-Taste vergrößert sich die Verfahrgeschwindigkeit, bei Betätigung der [OVRD ↓]-Taste verringert sie sich.
❿	12 Tasten für JOG-Betrieb: [–X/(J1)] ... [+C/(J6)]	Funktionstaste für JOG-Betrieb Im Gelenk-JOG-Betrieb können alle Gelenke einzeln bewegt werden. Im XYZ-JOG-Betrieb kann der Roboterarm an jeder der Koordinatenachsen entlang bewegt werden. Mit den Tasten erfolgt auch die Eingabe von Menüauswahlnummern oder Schrittnummern.

Tab. 2-3: *Übersicht der Bedienelemente der Teaching Box R32TB (1)*

Nr.	Bezeichnung	Funktion
⓫	[SERVO]-Taste	Betätigen Sie die [SERVO]-Taste bei halb durchgedrücktem Dreistufenschalter, um die Servoversorgungsspannung einzuschalten.
⓬	[MONITOR]-Taste	Wechselt in den Monitor-Modus und zeigt das Monitor-Menü an
⓭	[JOG]-Taste	Wechselt in den JOG-Modus und zeigt das JOG-Menü an
⓮	[HAND]-Taste	Wechselt in den Hand-Modus und zeigt das Hand-Menü an
⓯	[CHARACTER]-Taste	Ruft das Editier-Menü auf und wechselt z. B. beim Editieren von Positionsdaten zwischen Zahlen und Buchstaben
⓰	[RESET]-Taste	Quittierung eines Fehlercodes In Verbindung mit der [EXE]-Taste wird ein Programm zurückgesetzt.
⓱	[↑]-, [↓]-, [←]-, [→]-Taste	Bewegt den Cursor in die entsprechende Richtung
⓲	[CLEAR]-Taste	Löscht das Zeichen an der Cursor-Position
⓳	[EXE]-Taste	Dateneingabe oder Bewegung des Roboters im Direkt-Modus
⓴	Zeichentaste	Überschreibt das Zeichen an der Cursor-Position

Tab. 2-3: *Übersicht der Bedienelemente der Teaching Box R32TB (2)*

2.3 Betriebsrechte

Beim Anschluss mehrerer Geräte an das Steuergerät, z.B. Teaching Box und Personalcomputer, verfügt nur ein Gerät über die Betriebsrechte.

Zur Ausführung von Vorgängen, die den Roboter starten, z.B. ein Programmstart, benötigt ein Gerät die Betriebsrechte. Im Gegensatz dazu können alle Vorgänge, die den Roboter stoppen, z. B. ein Stopp-Befehl oder ein Ausschalten der Servoversorgung, aus Sicherheitsgründen auch ohne Betriebsrechte ausgeführt werden.

Schalter	**T/B [ENABLE/ DISABLE]**	**DISABLE**		**ENABLE**	
	Steuergerät [MODE]	**AUTOMATIC**	**MANUAL**	**AUTOMATIC**	**MANUAL**
Betriebsrechte	T/B	—	—	—②	●
	Steuergerät	●①	—	—②	—
	PC	●①	—	—②	—
	Externes Signal	●①	—	—②	—

Tab. 2-4: *Einstellung der Betriebsrechte*

① Erfolgt die Eingabe des Signals IOENA (Eingabe Betriebsrechte) über ein externes Gerät, besitzt das externe Signal die Betriebsrechte und die Betriebsrechte des PCs sind deaktiviert.

② Ist die Teaching Box auf „ENABLE“ gesetzt, erfolgt bei einer Einstellung des [MODE]-Schalters auf die Stellung „AUTOMATIC“ die Fehlermeldung „5000“.

In folgender Tabelle sind die Vorgänge aufgeführt, die ein Betriebsrecht erfordern:

Vorgang	Betriebsrecht erforderlich	Funktion
Operation	●	Servo EIN
	—	Servo AUS
	●	Programmstart
	—	Programmstopp/Zyklusstopp
	●	Anwendungsinitialisierung (Programm zurücksetzen)
	—	Alarm zurücksetzen
	●	Geschwindigkeitsübersteuerung ändern (ist über die T/B im Handbetrieb immer möglich)
	—	Einlesen der Geschwindigkeitsübersteuerung
	●	Programmnummer ändern
	—	Programm-/Zeilennummer lesen
Ein-/Ausgangs-signalfunktion	—	Eingangs-/Ausgangssignal lesen
	—	Ausgangssignal schreiben
	●	Spezielle Eingänge: Start, Reset, Servo EIN, Bremse EIN/AUS, manueller Modus-wechsel, allgemeinen Ausgang zurücksetzen, Programmnummer festlegen, Zei-lennummer festlegen, Geschwindigkeitsübersteuerung festlegen
	—	Spezielle Eingänge: Stopp, Servo AUS, kontinuierlicher Betrieb, Eingangssignal Betriebsrechte, Ausgabeanforderung Programmnummer, Ausgabeanforderung Zeilennummer, Ausgabeanforderung Geschwindigkeitsübersteuerung, Ausgabe-anforderung Fehlernummer/numerische Eingabe
	—	Handsensor-/Handsteuersignal lesen
	●	Handsteuersignal schreiben
Programm-editierung ①	—	Zeilennummer eingeben, lesen, aufrufen; Position hinzufügen, korrigieren, lesen; Variable schreiben, lesen
	●	Schrittweiterschaltung, Ausführung
	—	Vorwärts-/Rückwärtsschritt
	●	Sprung, direkte Ausführung, JOG-Betrieb
Dateifunktion	—	Programmverzeichnis lesen, Programm schützen/kopieren/löschen/umbenen-nen/zurücksetzen
Wartungsfunktion	—	Parameter lesen, Uhrzeit einstellen/lesen, Betriebszeit lesen, Alarmliste lesen
	●	Grundposition einstellen, Parameter ändern

Tab. 2-5: *Vorgänge und Betriebsrechte*

① Wird über ein Gerät eine Editierung online ausgeführt, ist zeitgleich keine Editierung über ein anderes Gerät möglich.

2.4 Bewegungs- und Steuerfunktionen

Das Steuergerät verfügt über folgende charakteristische Funktionen.

Funktion	Beschreibung	Referenz
Optimale Geschwindigkeit	Diese Funktion verhindert Fehler, die durch Geschwindigkeitsüberschreitungen hervorgerufen werden. Dazu werden bei einer Verfahrbewegung zwischen zwei Punkten Stellungen vermieden, die eine Geschwindigkeitsüberschreitung zur Folge hätten. Bei aktivierter Funktion ist die Geschwindigkeit an der Handspitze nicht konstant.	Abschn. 6.3.74 „Befehl Spd (Speed)“
Optimale Beschleunigung/ Verzögerung	Die Funktion legt die optimale Beschleunigungs-/Verzögerungszeit beim Starten und Stoppen des Roboters in Abhängigkeit der Lasteinstellungen fest. Zu den Lasteinstellungen zählen Gewicht, Abmessungen und Schwerpunkt von Hand und Werkstück. Die Funktion bewirkt eine Verkürzung der Zykluszeit.	Abschn. 6.3.56 „Befehl Mxt (Move External) Abschn. 6.3.47 „Befehl LoadSet (Load Set)“
XYZ-Weichheit	Die Funktion ermöglicht über die Daten des Servomotors eine nachgiebige Steuerung des Roboters. Die Funktion bewirkt ein sanftes Einsetzen von Werkstücken in Bohrungen o. Ä. Im kartesischen Koordinatensystem ist das Teachen von Positionen auch bei aktivierter Achsenweichheit möglich. Der sinnvolle Einsatz dieser Funktion ist von den Werkstückbedingungen abhängig.	Abschn. 6.3.10 „Befehl Cmp Tool (Compliance Tool)“
Kollisions-überwachung	Die Funktion bewirkt bei einem Zusammenstoß des Roboterarms mit umliegenden Einrichtungen einen sofortigen Stopp des Roboters. Dadurch können entstehende Schäden begrenzt werden. Ein Aktivierung der Funktion ist sowohl im Automatik- als auch im JOG-Betrieb möglich. Die Funktion kann nicht gemeinsam mit der Funktion zur Steuerung zusätzlicher Mechanismen verwendet werden.	Abschn. 6.3.14 „Befehl ColChk (Col Check)“
Überwachung der Wartungsintervalle	In Abhängigkeit des Roboterbetriebes erfolgt die Überwachung wartungsrelevanter Daten. Überwacht werden z. B. die Roboterbatterien, Zahnriemen oder Schmierstoffe. Die Wartungsinformationen können über die Programmier-Software angezeigt werden. Die Funktion kann nicht gemeinsam mit der Funktion zur Steuerung zusätzlicher Mechanismen verwendet werden.	Verwenden Sie die Software RT ToolBox2, um die Funktion zu nutzen.
Wiederherstellung von Positionsdaten	Die Funktion zur Wiederherstellung von Positionsdaten berechnet Korrekturwerte für die Nullpunkt-, die Werkzeug- und die Basiskoordinaten. Bei einer Abweichung der Gelenkachsendaten, durch Austausch eines Motors, durch Verformung eines Handgreifers oder einer Abweichung der Basiskoordinaten kann durch die Angabe von maximal 10 Punkten eine Korrektur der Positionsabweichung erfolgen. Die Funktion kann mit der Programmier-Software genutzt werden.	Verwenden Sie die Software RT ToolBox2, um die Funktion zu nutzen.
Kontinuierliche Bewegung	Die Funktion dient zur Steuerung einer Verfahrbewegung ohne Beschleunigung/Verzögerung zwischen verschiedenen Punkten. Sie bewirkt eine Verkürzung der Zykluszeit.	Abschn. 4.3.4 „Kontinuierliche Bewegung“ Abschn. 6.3.13 „Befehl Cnt (Continuous)“

Tab. 2-6: *Bewegungs- und Steuerfunktionen (1)*

Funktion	Beschreibung	Referenz
Multitasking	Die Multitasking-Funktion ermöglicht die parallele Ausführung mehrerer Programme zur Verkürzung der Taktzeiten. Der Roboter kann neben seiner Bewegung weitere Funktionen ausführen und mit der Peripherie kommunizieren, z. B. um Signale weiterzugeben. Mit Hilfe der Multitasking-Funktion kann über ein Programm eine Steuerung peripherer Einrichtungen ohne Einsatz einer SPS erfolgen.	Abschn. 4.11 „Multitasking-Funktion“ Abschn. 6.3.86 „Befehl XRun (X Run)“
Ständige Programmausführung	Die Funktion ermöglicht die Ausführung eines Programmes, sobald die Versorgungsspannung eingeschaltet wird. Mit Hilfe der Funktion kann im Multitasking-Betrieb über ein Programm eine SPS simuliert werden.	Abschn. 9.5 „Parameter SLT (ALWAYS)“
Programm fortsetzen	Für den Programmplatz 1 wird nach dem Ausschalten der Spannungsversorgung die aktuelle Position innerhalb der Anwendung gespeichert. Nach dem nächsten Einschalten der Spannungsversorgung startet die Anwendung von dieser gespeicherten Position.	Abschn. 9.5 „Parameter CTN“
Steuerung einer Zusatzachse	Die Funktion ermöglicht eine Steuerung von bis zu 2 Zusatzachsen. Da die Positionen der Achsen in den geteachten Roboterdaten gespeichert sind, kann eine vollständig synchrone Steuerung der Achsen erfolgen. Neben der Bewegung der Zusatzachsen ist eine Bewegung über Kreis-Interpolation möglich.	Handbuch „Schnittstelle zur Steuerung einer Zusatzachse“
Steuerung zusätzlicher Mechanismen	Die Funktion ermöglicht neben den Standard-Robotern eine Steuerung von bis zu 2 zusätzlichen Mechanismen, die über Servomotoren angetrieben werden.	Handbuch „Schnittstelle zur Steuerung einer Zusatzachse“
Kommunikation	Zur Kommunikation mit externen Einheiten stehen folgende Möglichkeiten zur Verfügung: Steuerung des Steuergerätes und Programmsteuerung • über Ein-/Ausgangssignale CRnD: parallele Schnittstelle mit max. 256/256 Ein-/Ausgängen CRnQ: max. 8192/8192 Ein-/Ausgänge bei SPS-Anschluss • über CC-Link-Netzwerk (optional) Als Datenverbindung • über RS232C-Schnittstelle (1 Standardschnittstelle) • über Ethernet Die Datenverbindung bezieht sich auf definierte Funktionen zum Austausch von Daten, z.B. Kompensationsdaten, mit externen Einheiten (z.B. optische Sensoren).	Abschn. 7.2.28 „Variablen M_IN, M_INB, M_INW“ Abschn. 7.2.36 „Variablen M_OUT, M_OUTB, M_OUTW“ Abschn. 9.16 „Kommunikationseinstellungen“
Interrupt-Überwachung	Die Funktion dient zur Überwachung von Signalen während der Programmabarbeitung. Bedingungsabhängig kann das Programm zur Ausführung einer Interrupt-Routine unterbrochen werden. Die Funktion wird z. B. verwendet, um zu überwachen, dass beim Transport von Werkstücken keine Werkstücke verloren gehen.	Abschn. 6.3.19 „Befehl Def Act (Define Act)“ Abschn. 6.3.2 „Befehl Act (Act)“
Unterprogrammaufruf	Die Funktion dient zum Aufruf eines Unterprogramms aus einem Hauptprogramm.	Abschn. 6.3.4 „Befehl CallP (Call P)“

Tab. 2-6: *Bewegungs- und Steuerfunktionen (2)*

Funktion	Beschreibung	Referenz
Palettierung	Die Funktion berechnet automatisch die Zwischenpositionen von in einer Palette angeordneten Werkstücken. Dadurch wird der Aufwand an zu teachenden Positionen minimiert. Die Funktion ermöglicht die Definition von Paletten im Spalten- und Zeilenformat sowie kreisförmigen Paletten.	Abschn. 4.4 „Palettierung“ Abschn. 6.3.26 „Befehl Def Plt (Define Pallet) Abschn. 6.3.63 „Befehl Plt (Pallet)“
Benutzerdefinierter Bereich	Die Funktion ermöglicht eine Echtzeit-Überwachung des Roboters in einem über max. 32 Flächen frei definierbaren Bereich. Befindet sich die Handspitze des Roboters innerhalb dieses Bereichs, kann die Ausgabe eines Statussignals an eine externe Einheit erfolgen, die dann in einem Programm weiter verarbeitet wird oder eine Fehlermeldung zur Folge hat. Zusätzlich stehen 2 weitere ähnliche Funktionen für ein Roboterprogramm (ZONE und ZONE2) zur Verfügung.	Abschn. 9.9 „Benutzerdefinierter Bereich“ Abschn. 8.2.53 „Funktion ZONE“ Abschn. 8.2.54 „Funktion ZONE2“ Abschn. 8.2.55 „Funktion ZONE3“
Verfahrweggrenzen für Gelenkbewegungen Verfahrweggrenzen für XYZ-Bewegungen Frei definierbare Begrenzungsfläche	Folgende Funktionen ermöglichen eine Begrenzung des Roboter-Arbeitsbereichs: Bei Gelenkbewegungen: Legt die Verfahrweggrenzen für jedes einzelne Gelenk fest. Bei XYZ-Bewegungen: Legt die Verfahrweggrenzen für das XYZ-Koordinatensystem fest. Frei definierbare Begrenzungsfläche: Der Arbeitsbereich eines Roboters kann über eine frei definierbare Fläche auf den Bereich vor oder hinter dieser Fläche begrenzt werden.	Kap. 9 „Parameter MEJAR“ und „Parameter MEPAR“ Abschn. 9.10 „Verfahrwegbegrenzungsebene“

Tab. 2-6: *Bewegungs- und Steuerfunktionen (3)*

3 Bedienung und Programmierung

In diesem Abschnitt werden die Bedienung der Teaching Box und die Funktionen der einzelnen Menüs beschrieben.

3.1 Menübaum

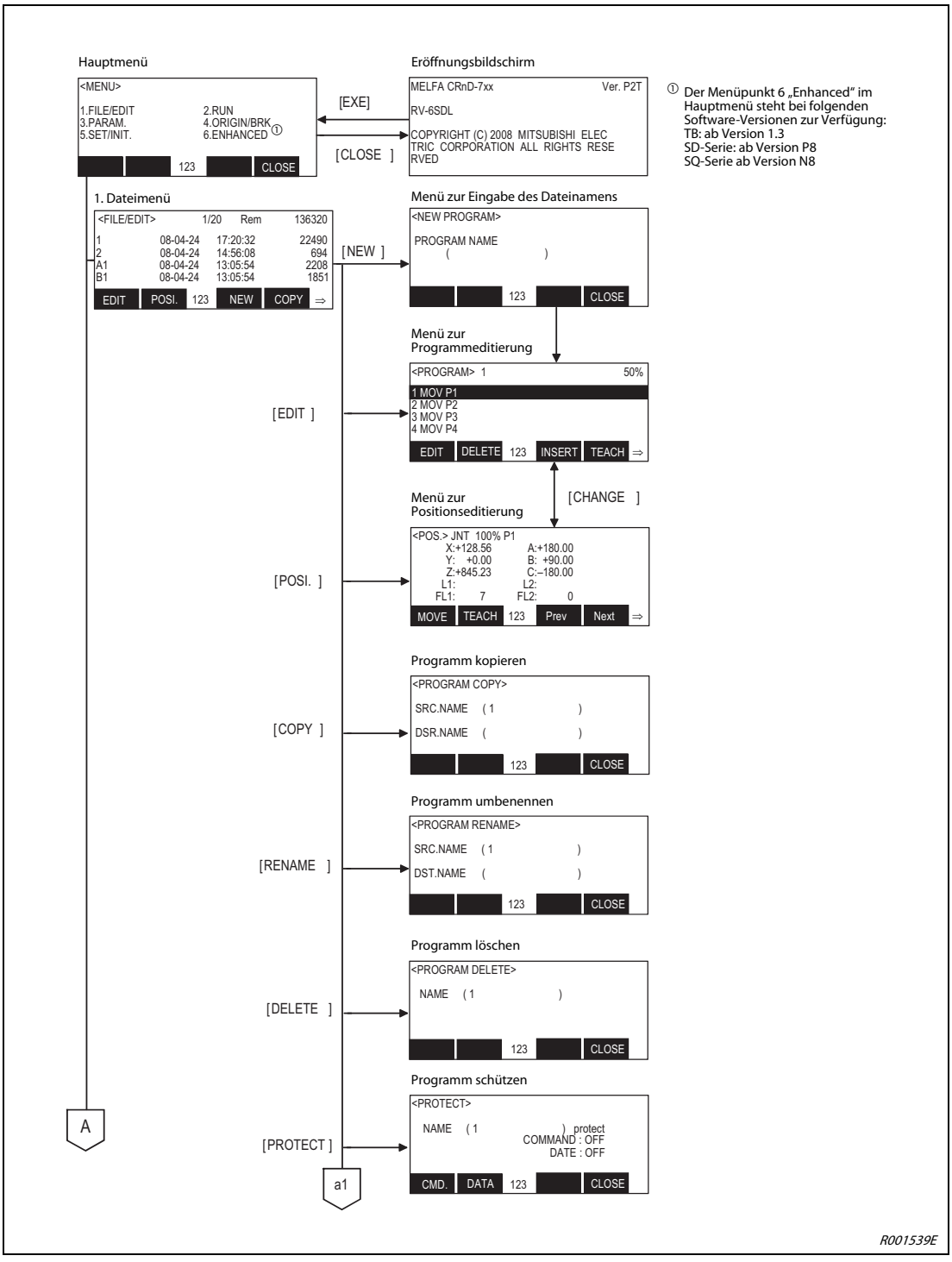


Abb. 3-1: *Menübaum (1)*

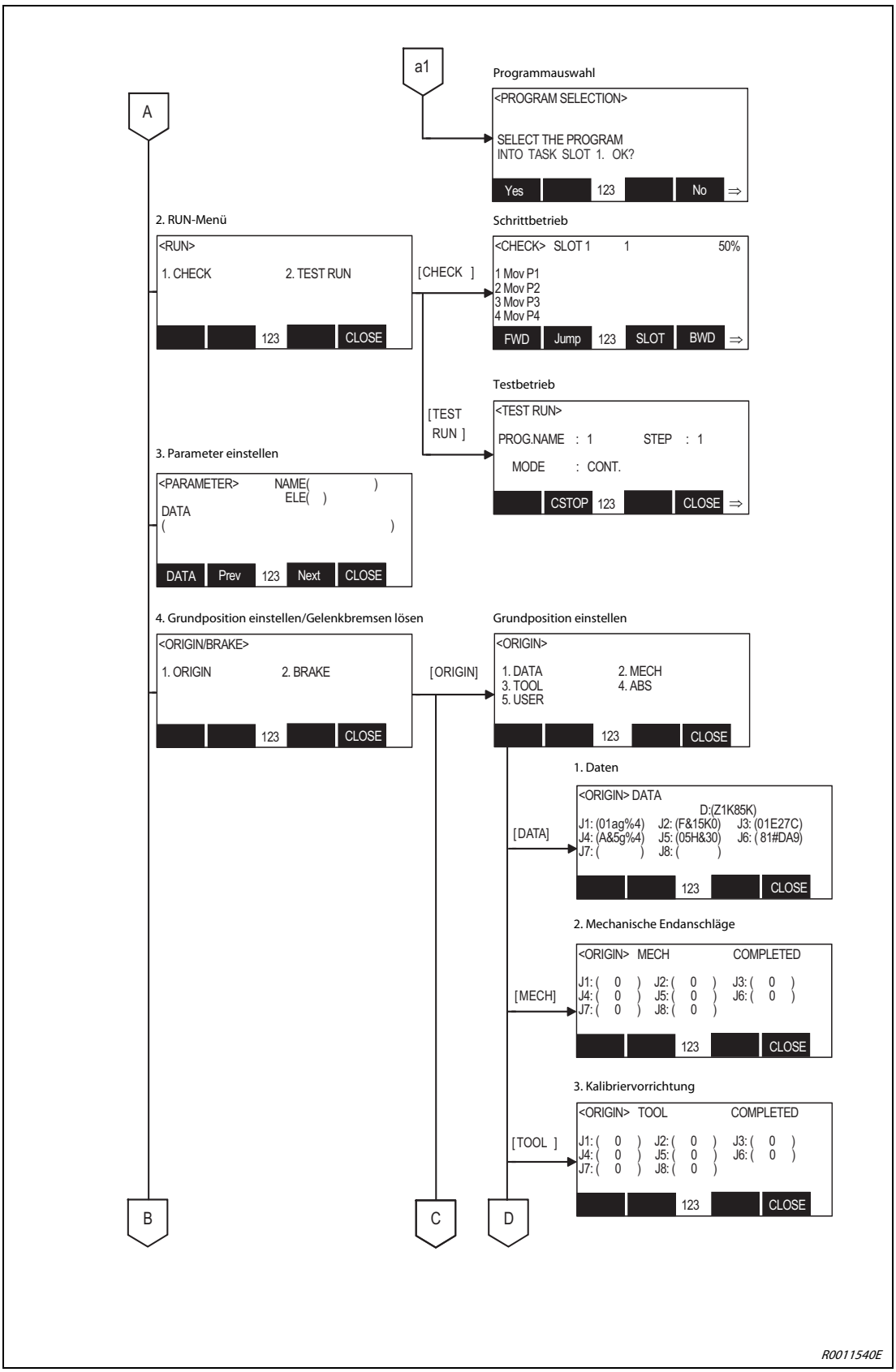


Abb. 3-1: *Menübaum (2)*

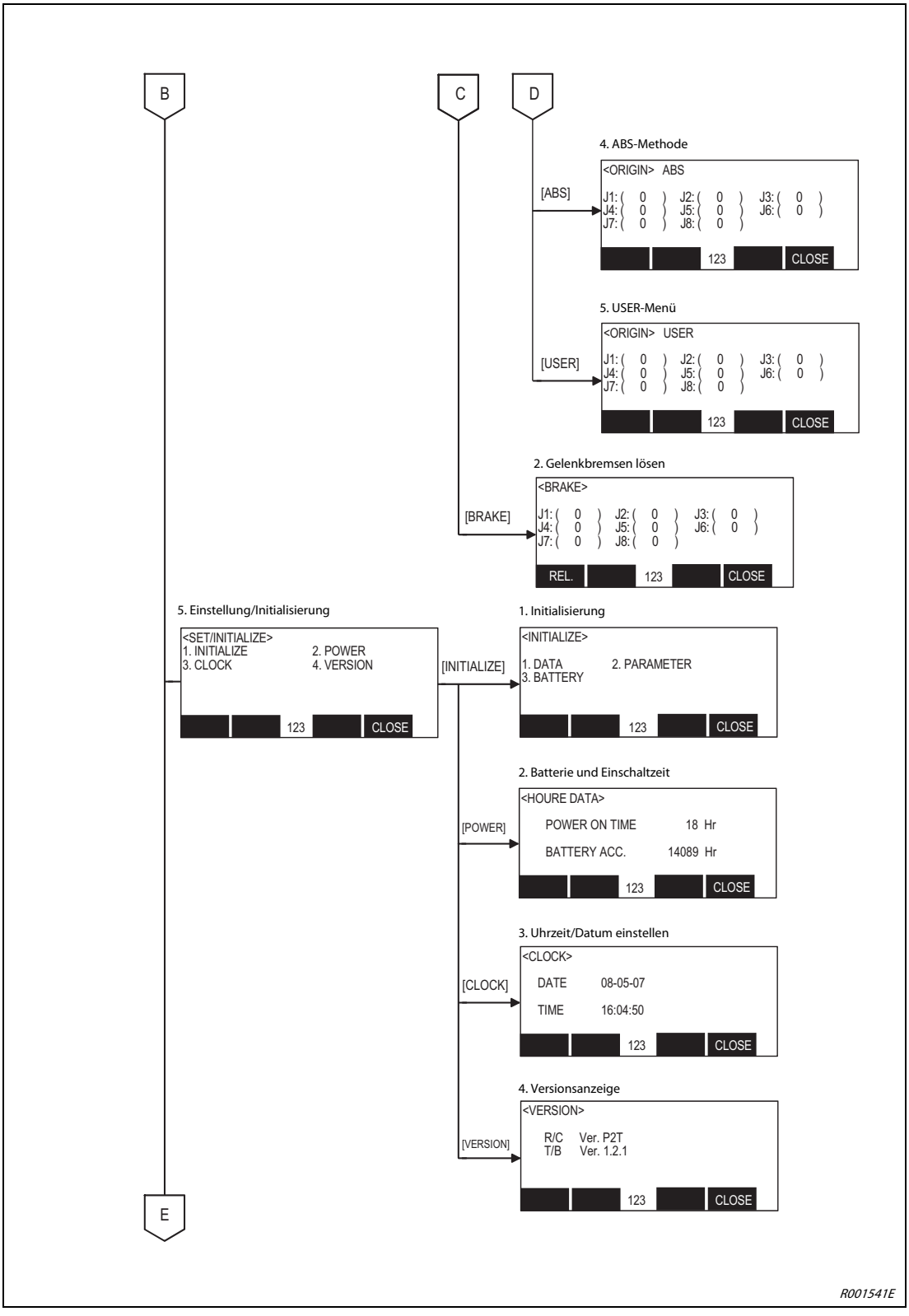


Abb. 3-1: *Menübaum (3)*

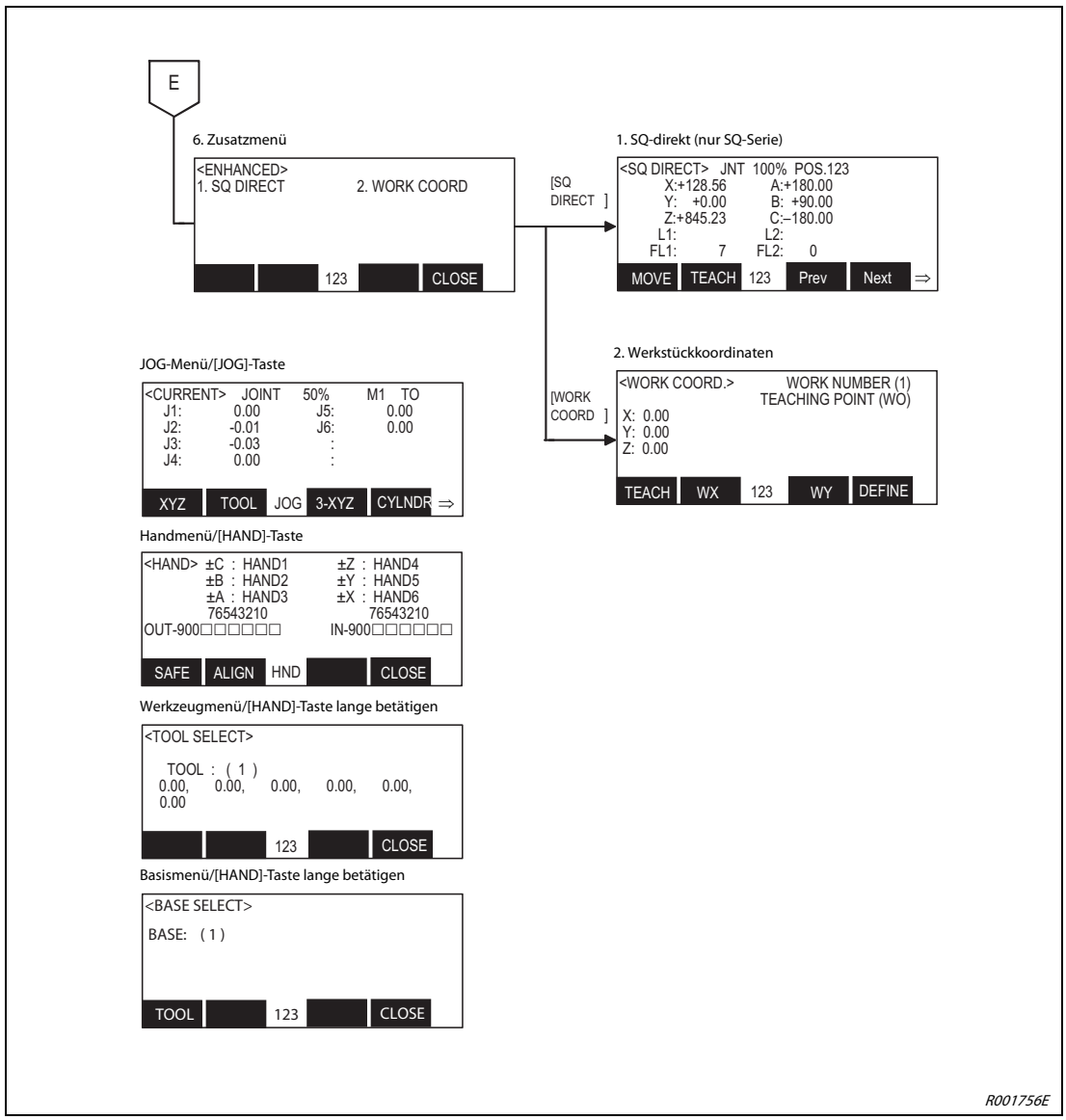


Abb. 3-1: *Menübaum (4)*

3.2 Eingabe eines Zeichens

Bei jeder Betätigung der [CHARACTER]-Taste wechselt der Schreibmodus zwischen der Zahlen- und Buchstabeneingabe. Der aktuelle Modus wird unten, in der Mitte des Displays angezeigt.

Eingabe von Zahlen

Die Eingabe von Zahlen erfolgt im Zahlenmodus über die Tasten, auf denen links unten die entsprechende Zahl sowie das Minus-Zeichen und das Komma angegeben sind.

Beispiel▽ Die Zahl „51“ wird als Programmnamen eingegeben.
Betätigen Sie dazu die [CHARACTER]-Taste und die Tasten [5] und [1].

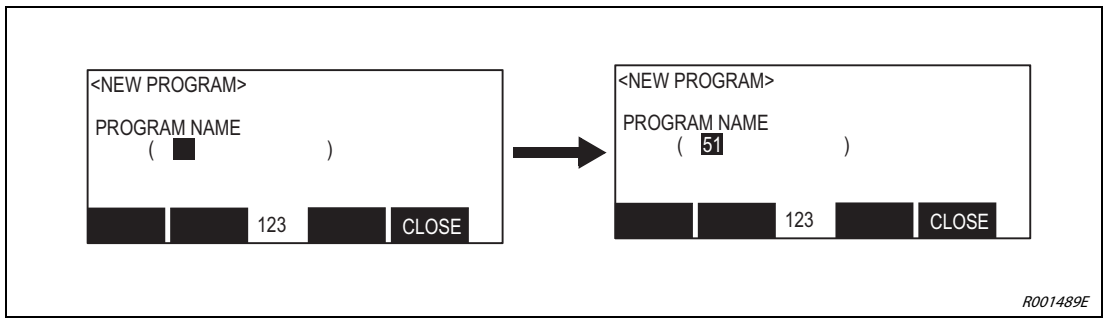


Abb. 3-2: *Eingabe von Zahlen*

△

Eingabe von Buchstaben

Die Eingabe von Buchstaben erfolgt im Buchstabenmodus über die Tasten, auf denen rechts unten der entsprechende Buchstabe angegeben ist. Bei mehrmaliger Betätigung der Taste wechselt das Zeichen. So erfolgt zum Beispiel bei mehrmaliger Betätigung der Taste [ABC] die Auswahl folgender Zeichen: „A“ ... „B“ ... „C“ ... „a“ ... „b“ ... „c“.
Mit Hilfe der Pfeiltaste [→] kann der Cursor bei der Auswahl von Buchstaben, die auf derselben Taste liegen, eine Stelle weiter bewegt werden.

Beispiel▽ Die Buchstaben „ABY“ sollen eingegeben werden.
Betätigen Sie dazu folgende Tasten: 1 × [ABC], [→], 2 × [ABC], 3 × [WXYZ].

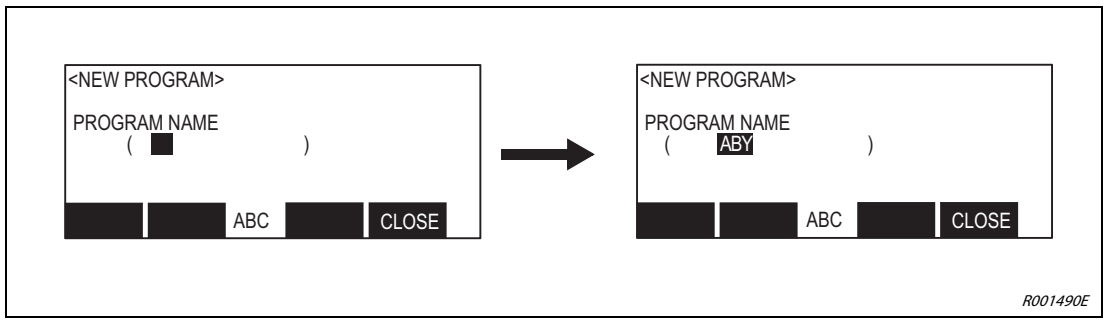


Abb. 3-3: *Eingabe von Buchstaben*

△

Die Tasten sind mit folgenden Zeichen belegt:

- [’ ( )]-Taste: ’ → (→) → “ → ^ → : → ; → ¥ → ?
- [@ =]-Taste: @ → = → + → − → ∗ → / → < → >
- [, %]-Taste: , → % → # → $ → ! → & → _ → .

Löschen eines Zeichens

Löschen Sie ein fehlerhaft eingegebenes Zeichen, indem Sie den Cursor auf dem Zeichen platzieren und die [CLEAR]-Taste betätigen.

Beispiel ▽ Der Buchstabe „B" der Zeichenfolge „ABY" soll in „M" geändert werden, so dass die Zeichenfolge „AMY" entsteht.
Bewegen Sie dazu den Cursor mit der [←]-Taste auf das Zeichen „B" und betätigen Sie folgende Tasten: [CLEAR], 1 × [MNO].

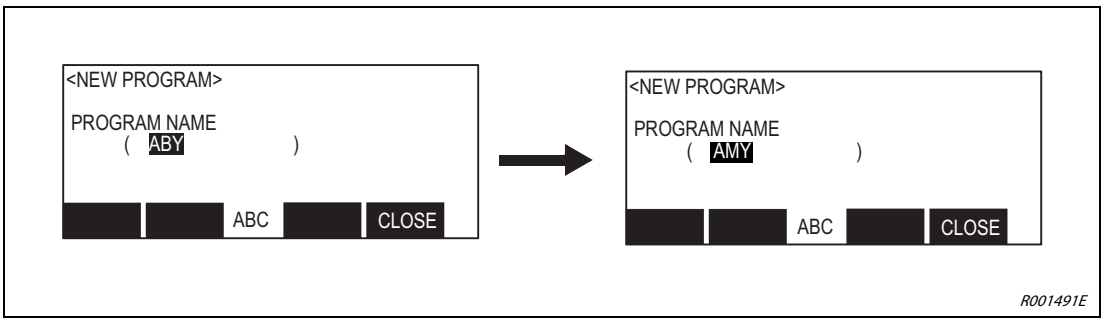


Abb. 3-4: *Löschen eines Zeichens*

△

HINWEIS Bei einer längeren Betätigung der [CLEAR]-Taste werden alle Zeichen in der Klammer gelöscht.

3.3 Auswahl eines Menüpunkts

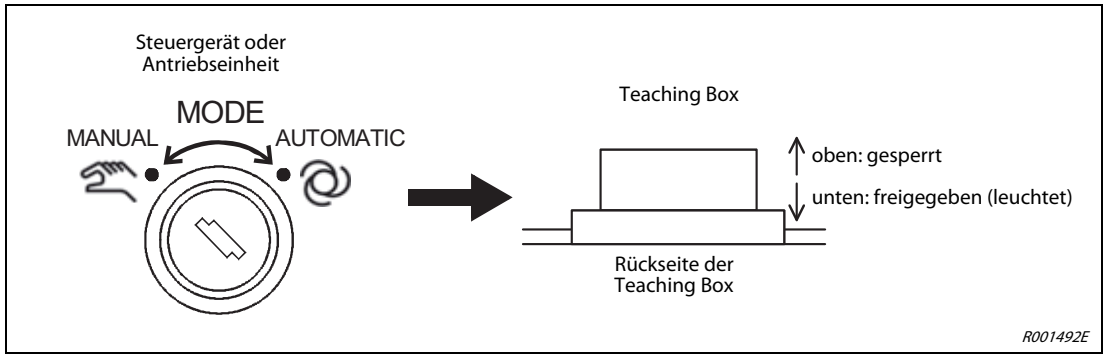
Es gibt zwei Möglichkeiten ein Menü aufzurufen:

- Menüauswahl über Eingabe einer Nummer
- Menü mit dem Cursor auswählen und [EXE]-Taste betätigen

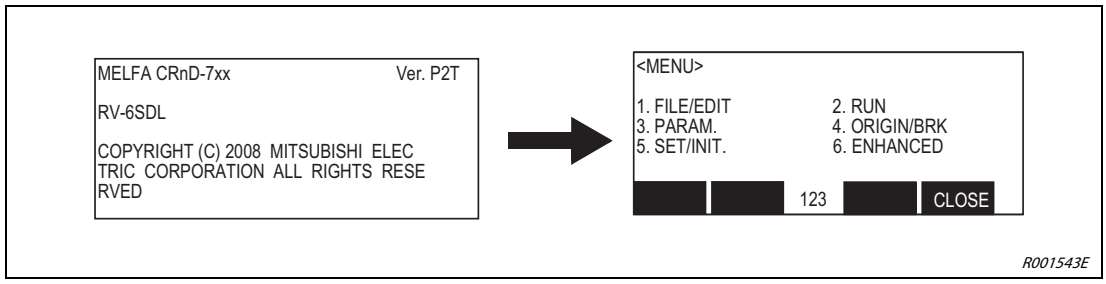
Ausführung

Nachfolgend werden die beiden Möglichkeiten beispielhaft an der Auswahl des Menüpunkts „1. FILE/EDIT“ gezeigt.

① Stellen Sie den [MODE]-Schalter des Steuergerätes auf die Stellung „MANUAL“. Aktivieren Sie die Teaching Box, indem Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „freigegeben“ stellen.

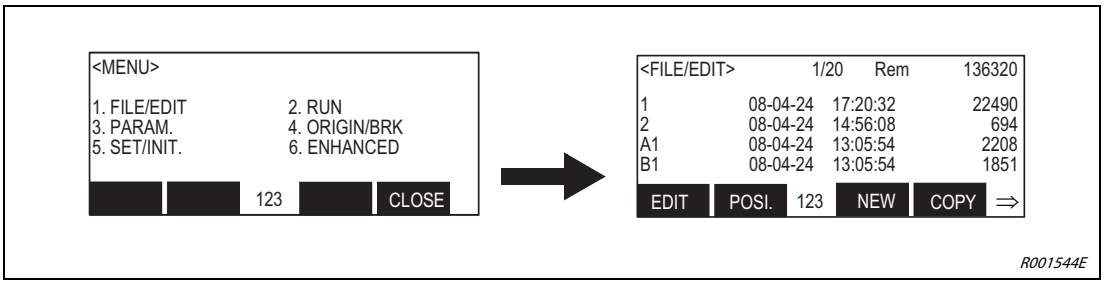


② Nach dem Einschalten erscheint der Eröffnungsbildschirm. Betätigen Sie nach Erscheinen des Eröffnungsbildschirms die Taste [EXE], um das Hauptmenü aufzurufen.



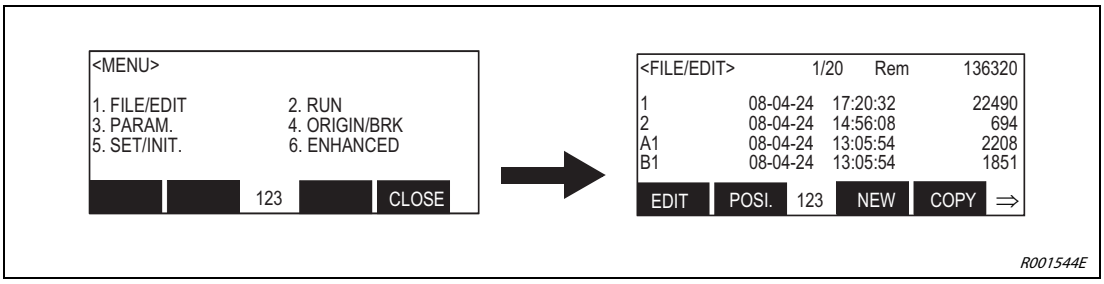
- Menüauswahl über Eingabe einer Nummer

① Wählen Sie das Menü „FILE/EDIT“ durch Eingabe der Ziffer „1“ aus. Das Menü „FILE/EDIT“ wird angezeigt.



- Menü mit dem Cursor auswählen und [EXE]-Taste betätigen

① Bewegen Sie den Cursor über die Pfeiltasten zum Menüpunkt „FILE/EDIT“ und betätigen Sie die EXE]-Taste. Das Menü „FILE/EDIT“ wird angezeigt.



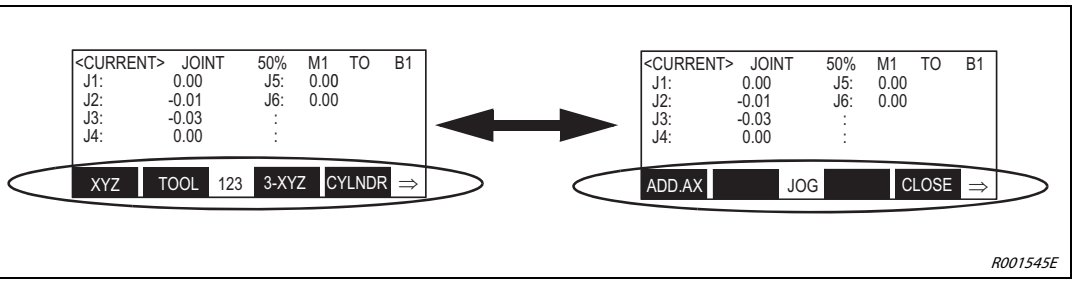
HINWEIS

Solange der [MODE]-Schalter des Steuergerätes nicht auf „MANUAL“ gestellt ist, sind über die ausgeschaltete Teaching Box nur bestimmte Funktionen (z. B. Anzeige der aktuellen Position im JOG-Betrieb, Änderung der Geschwindigkeitsübersteuerung, Anzeige der Ein- und Ausgangssignalzustände, Fehlerliste usw.) ausführbar.

Funktionstasten

Die Menüs in der untersten Zeile der Anzeige sind den Funktionstasten [F1], [F2], [F3] und [F4] zugewiesen. Wählen Sie das Menü durch Betätigung der korrespondierenden Funktionstaste.

Die Anzeige „=>“ bedeutet, dass das Menü auf einer nächsten Seite fortgesetzt wird. Betätigen Sie die entsprechende Funktionstaste, um das Menü weiterzuschalten.

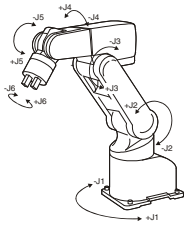
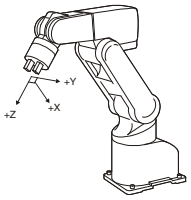
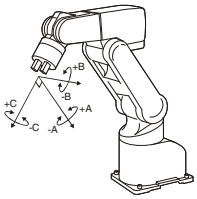
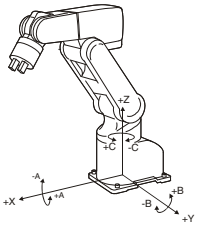


3.4 Roboter im JOG-Betrieb bewegen

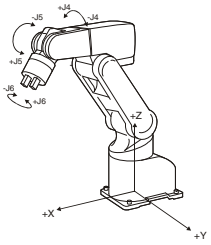
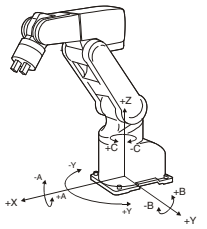
Im JOG-Betrieb kann der Roboter schrittweise manuell positioniert werden. In diesem Abschnitt wird der JOG-Betrieb anhand eines 6-achsigen Vertikal-Knickarmroboters erläutert. Die Achsenkonfiguration ist abhängig vom verwendeten Robotertyp. Eine detaillierte Beschreibung zu den einzelnen Robotertypen finden Sie im Technischen Handbuch des jeweiligen Roboters.

3.4.1 JOG-Betriebsarten

Es werden 6 JOG-Betriebsarten unterschieden:

Betriebsart	Betrieb	Beschreiben
Gelenk-JOG-Betrieb  R000862C	• Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf die Stellung „ENABLE“. • Halten Sie den Dreistufenschalter in Mittelstellung. • Betätigen Sie die [SERVO]-Taste. (Die Servospannung wird eingeschaltet.) • Betätigen Sie die [JOG]- und die [F1]-Taste, um in den Gelenk-JOG-Betrieb zu wechseln. • Betätigen Sie zur Bewegung der Gelenke die entsprechende Taste J1 bis J6.	Im Gelenk-JOG-Betrieb können die Roboterachsen einzeln verfahren werden. Dabei ist eine unabhängige Einstellung der Achsen J1 bis J6 und der Zusatzachsen J7 und J8 möglich. Die Anzahl der Achsen hängt vom Robotertyp ab. Die Steuerung der Zusatzachsen J7 und J8 erfolgt über die Tasten [J1] und [J2].
Werkzeug-JOG-Betrieb  R000863C  R000864C	Führen Sie die oben genannten ersten drei Punkte aus. • Betätigen Sie die Funktionstaste, um in den Werkzeug-JOG-Betrieb zu wechseln. • Betätigen Sie zur Bewegung der Achsen die entsprechende Taste X, Y, Z, A, B, C.	Im Werkzeug-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im Werkzeug-Koordinatensystem bewegt werden. Die Handspitze wird linear bewegt. Die Stellung des Roboters kann über die Tasten A, B und C um die Achsen X, Y, und Z des Werkzeug-Koordinatensystems gedreht werden, ohne die Position der Handspitze zu verändern. Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden. Das Werkzeug-Koordinatensystem, in dem die Position der Handspitze festgelegt wird, ist vom Robotertyp abhängig. Beim Vertikal-Knickarmroboter ist die Richtung vom Handflansch zur Handspitze als +Z definiert. Beim SCARA-Roboter ist die Richtung von der Aufstellfläche nach oben als +Z definiert.
XYZ-JOG-Betrieb  R000865C	Führen Sie die oben genannten ersten drei Punkte aus. • Betätigen Sie die Funktionstaste, um in den XYZ-JOG-Betrieb zu wechseln. • Betätigen Sie zur Bewegung der Achsen die entsprechende Taste X, Y, Z, A, B, C.	Im XYZ-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im XYZ-Koordinatensystem bewegt werden. Die Stellung des Roboters kann über die Tasten A, B und C um die Achsen X, Y, und Z des XYZ-Koordinatensystems gedreht werden, ohne die Position der Handspitze zu verändern. Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden.

Tab. 3-1: *JOG-Betriebsarten (1)*

Betriebsart	Betrieb	Beschreiben
3-Achsen-XYZ-JOG-Betrieb  *R000866C*	Führen Sie die oben genannten ersten drei Punkte aus. ● Betätigen Sie zweimal die Funktionstaste, um in den 3-Achsen-XYZ-JOG-Betrieb zu wechseln. ● Betätigen Sie zur Bewegung der Achsen die entsprechende Taste X, Y, Z, J4, J5, J6.	Im 3-Achsen-XYZ-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im XYZ-Koordinatensystem bewegt werden. Im Unterschied zum XYZ-JOG-Betrieb wird die Stellung des Roboters wie im Gelenk-JOG-Modus durch Drehung der Achsen J4, J5 und J6 verändert. Bei fest definierter Position der Handspitze wird die Stellung über die Achsen X, Y, Z, J4, J5 und J6 interpoliert, d. h. die Stellung ist nicht konstant. Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden.
Kreis-JOG-Betrieb  *R000867C*	Führen Sie die oben genannten ersten drei Punkte aus. ● Betätigen Sie dreimal die Funktionstaste, um in den Kreis-JOG-Betrieb zu wechseln. ● Betätigen Sie zur Bewegung der Achsen die entsprechende Taste X, Y, Z, A, B, C.	Im Kreis-JOG-Betrieb kann die Position der Handspitze kreisförmig um den Nullpunkt bewegt werden. Eine Änderung der X-Achsen-Koordinate bewirkt vom Mittelpunkt des Roboters ausgehend eine radiale Bewegung der Handspitze. Eine Änderung der Y-Achsen-Koordinate bewirkt die gleiche Bewegung wie die Steuerung der J1-Achse im Gelenk-JOG-Betrieb. Eine Änderung der Z-Achsen-Koordinate bewirkt eine Bewegung der Hand in Z-Richtung wie beim XYZ-JOG-Betrieb. Bei einer Änderung der Koordinaten der A-, B- oder C-Achse erfolgt eine Drehung des Handgreifers wie im XYZ-JOG-Betrieb. Die Achsen sind bei Robotern vom Typ RH steuerbar.
Werkstück-JOG-Betrieb *R001546E*	Führen Sie die oben genannten ersten drei Punkte aus. ● Betätigen Sie dreimal die Funktionstaste, um in den Werkstück-JOG-Betrieb zu wechseln. ● Betätigen Sie zur Bewegung der Achsen die entsprechende Taste X, Y, Z, A, B, C.	Im Werkstück-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im Werkstück-Koordinatensystem bewegt werden. Die Stellung des Roboters kann über die Tasten A, B und C um die Achsen X, Y, und Z des Werkstück-Koordinatensystems gedreht werden, ohne die Position der Handspitze zu verändern. Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden. HINWEISE: Werkstück-Koordinatensystem: Legen Sie das Werkstück-Koordinatensystem vor dem JOG-Betrieb fest. (Es stehen 8 Parameter (WKnCORD, n = 1 bis 8) zur Einstellung des Koordinatensystems zur Verfügung.) Ist das Werkstück-Koordinatensystem nicht eingestellt, wird der Roboter im XYZ-JOG-Betrieb bewegt. Weitere Informationen finden Sie im Technischen Handbuch des Roboters. Die Funktion steht in folgenden Software-Versionen zur Verfügung: TB: ab Version 1.3 SD-Serie: ab Version P8 SQ-Serie ab Version N8

Tab. 3-1: *JOG-Betriebsarten (2)*

HINWEIS

Nähert sich der Überwachungspunkt der Hand in einer der in Tab. 3-1 beschriebenen JOG-Betriebsarten einem singulären Punkt, erscheint ein Warnsymbol auf der Teaching Box und es ertönt ein Warnton. Die Funktion kann über den Parameter MESNGLSW deaktiviert werden. Eine detaillierte Beschreibung der Parameter finden Sie in Kap. 9. Eine Beschreibung der Funktion „Fehlermeldung bei Erreichen des singulären Punkts“ finden Sie in Abschn. 9.20.

3.4.2 JOG-Geschwindigkeit einstellen

Die Geschwindigkeit wird in % angezeigt. Wenn Sie diesen Wert ändern wollen, betätigen Sie die [OVRD ↑]- oder die [OVRD ↓]-Taste. Folgende Geschwindigkeiten können eingestellt werden:

[OVRD ↑]-Taste								[OVRD ↓]-Taste
LOW	HIGH	3 %	5 %	10 %	30 %	50 %	70 %	100 %

Die Einstellungen „LOW“ und „HIGH“ sind vordefinierte Werte. Bei diesen Einstellungen wird der Roboter bei jedem Tastendruck um einen bestimmten Verfahrweg weiterbewegt. Die Größe des Verfahrwegs hängt vom Robotertyp ab.

Schritte/Tastendruck	JOG-Betriebsart	
	Gelenk	Werkzeug/XYZ
LOW	0,01°	0,01 mm
HIGH	0,10°	0,10 mm

Tab. 3-2: *Geschwindigkeitswerte für die einzelnen JOG-Betriebsarten beim Roboter RV-6SD*

3.4.3 Gelenk-JOG-Betrieb

Im Gelenk-JOG-Betrieb kann jede Roboterachse in Winkelgraden einzeln verfahren werden.

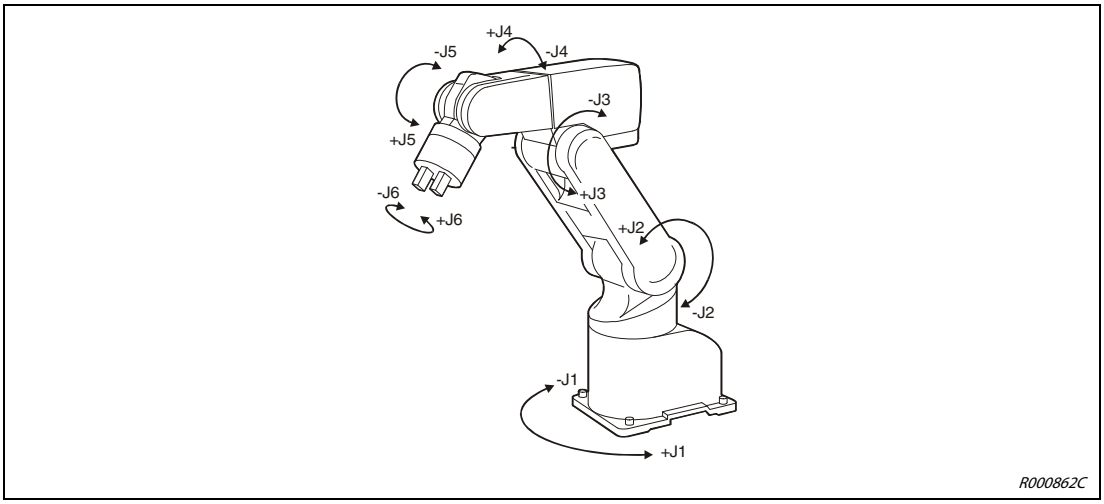


Abb. 3-5: *Bewegungsrichtungen des Roboters im Gelenk-JOG-Betrieb*

3.4.4 Werkzeug-JOG-Betrieb

Im Werkzeug-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im Werkzeug-Koordinatensystem bewegt werden. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Orientierungsdaten A, B und C erfolgt in Grad.

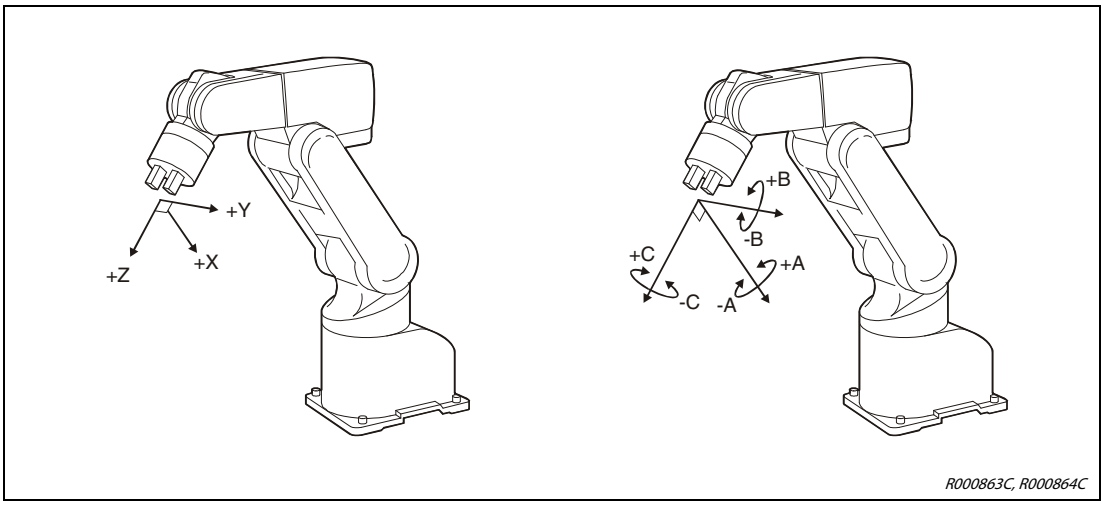


Abb. 3-6: *Bewegungsrichtungen des Roboters im Werkzeug-JOG-Betrieb*

3.4.5 XYZ-JOG-Betrieb

Im XYZ-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im XYZ-Koordinatensystem bewegt werden. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Orientierungsdaten A, B und C erfolgt in Grad.

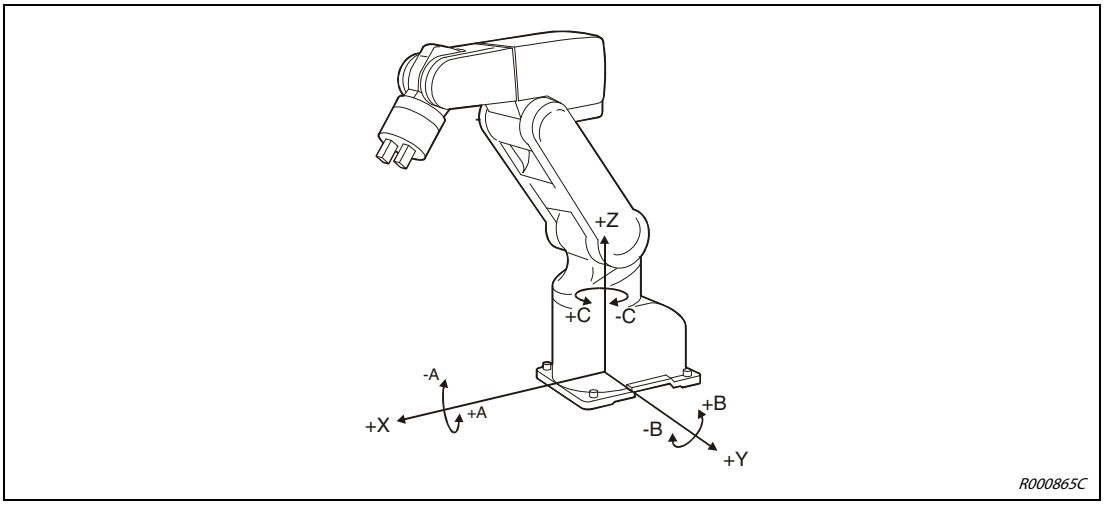


Abb. 3-7: *Bewegungsrichtungen des Roboters im XYZ-JOG-Betrieb*

3.4.6 3-Achsen-XYZ-JOG-Betrieb

Im 3-Achsen-XYZ-JOG-Betrieb erfolgt die Änderung der Koordinaten für die X-, Y- und Z-Achse wie im XYZ-JOG-Betrieb. Unabhängig davon erfolgt eine Änderung der Gelenkdaten wie im Gelenk-JOG-Betrieb, wobei die Position des Überwachungspunktes der Hand (X-, Y- und Z-Wert) durch Änderungen der Stellung aufrecht erhalten wird. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Gelenkdaten J4, J5 und J6 erfolgt in Grad.

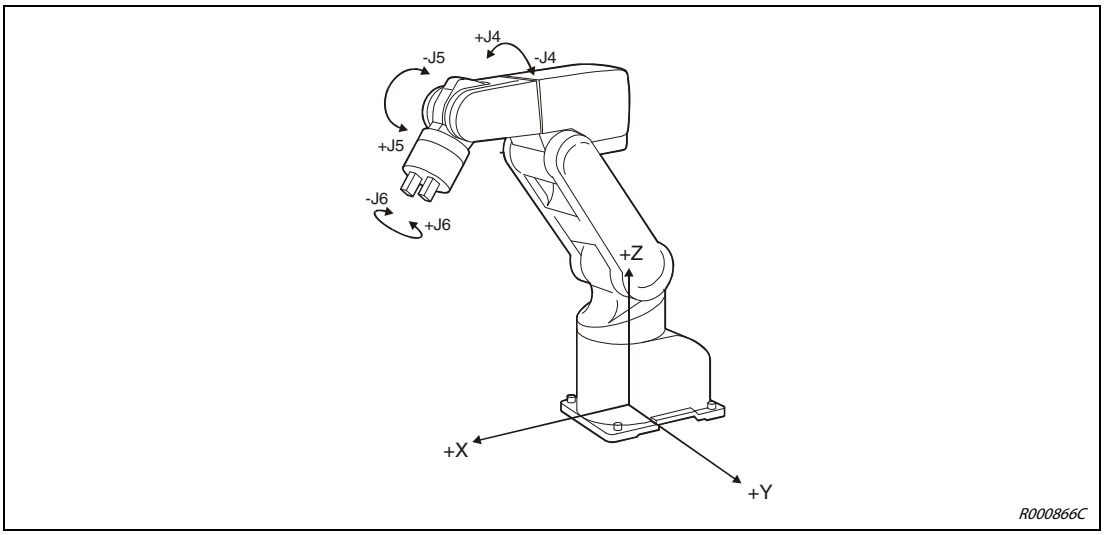


Abb. 3-8: *Bewegungsrichtungen des Roboters im 3-Achsen-XYZ-JOG-Betrieb*

3.4.7 Kreis-JOG-Betrieb

Eine Änderung der X-Achsen-Koordinate bewirkt vom Mittelpunkt des Roboters ausgehend eine radiale Bewegung der Handspitze. Eine Änderung der Y-Achsen-Koordinate resultiert in einer Drehung um die J1-Achse. Eine Änderung der Z-Achsen-Koordinate bewirkt eine Bewegung der Hand entlang der Z-Achse. Bei einer Änderung der Koordinaten der A-, B- oder C-Achse erfolgt eine Drehung des Handgreifers wie im XYZ-JOG-Betrieb. Die Einstellung der Koordinaten X und Z erfolgt in mm, die Einstellung der Daten Y, A, B und C erfolgt in Grad.

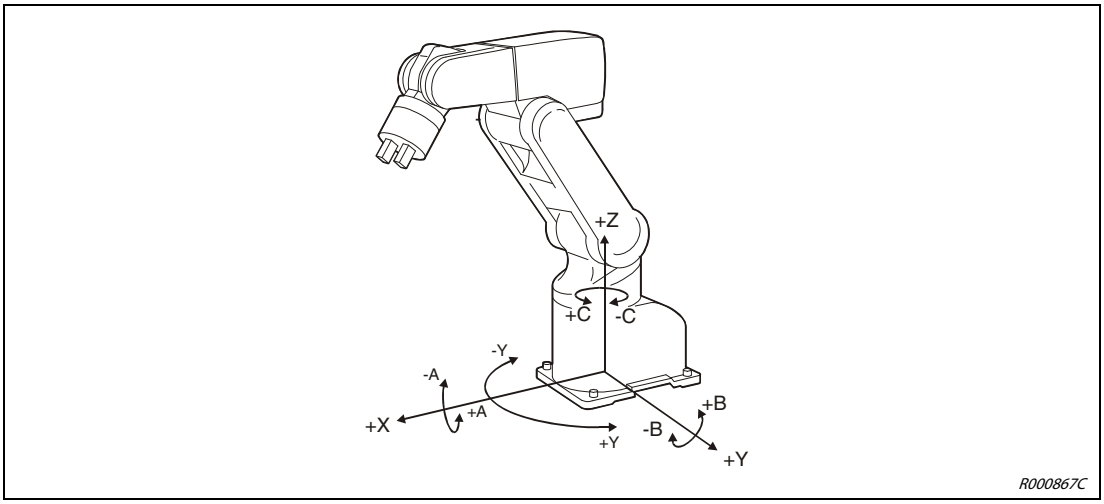
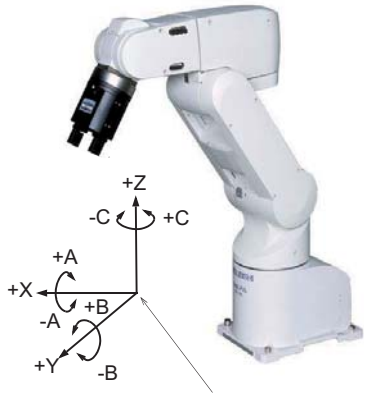


Abb. 3-9: *Bewegungsrichtungen des Roboters im XYZ-JOG-Betrieb*

3.4.8 Werkstück-JOG-Betrieb

Im Werkstück-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im Werkstück-Koordinatensystem bewegt werden. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Orientierungsdaten A, B und C erfolgt in Grad.



Werkstück-Koordinatensystem:
Koordinatensystem, das am Werkstück, dem Werktisch o. Ä. ausgerichtet ist.

R001547E

Abb. 3-10: *Bewegungsrichtungen des Roboters im Werkstück-JOG-Betrieb*

3.4.9 Kollisionsüberwachung im JOG-Betrieb

Die Kollisionsüberwachung kann über einen Parameter aktiviert werden. Erfasst das Steuergerät einen Zusammenstoß des Roboters mit einer umliegenden Einrichtung, erfolgt die Ausgabe der Fehlernummer 101n. (Die letzte Stelle „n" gibt die Achsennummer wieder.) Auch während des JOG-Betriebs ist eine Aktivierung dieser Funktion möglich. Ob die Funktion standardmäßig aktiviert ist, hängt vom Robotermodell ab.

Parameter		Anzahl der Zeichen	Beschreibung	Werkseinstellung
Kollisionsüberwachung ①	COL	Ganze Zahl 3	Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung 1. Element: Kollisionsüberwachung 0 = deaktiviert 1 = aktiviert 2. Element: Aktivierung direkt nach Einschalten der Spannungsversorgung 0 = deaktiviert 1 = aktiviert 3. Element: Freigabe im JOG-Betrieb 0 = deaktiviert 1 = aktiviert 2 = NOERR-Modus Im NOERR-Modus erfolgt keine Fehlerausgabe, auch wenn die Kollisionsüberwachung anspricht. Die Servoversorgung wird jedoch abgeschaltet. Verwenden Sie diesen Modus, wenn kein störungsfreier Betrieb durch häufiges Ansprechen der Kollisionsüberwachung möglich ist. Befindet sich der Roboter nicht im Programmbetrieb (das gilt auch für die Anfahrt einer Position und den Schrittbetrieb), hängt die Funktion von der Einstellung für den JOG-Betrieb ab (3. Element).	Für die Robotermodelle RV-SD/SQ: 0,0,1 RH-SDH/SQH: 1,0,1 RH-SDHR/SQHR: 1,1,1
Ansprechschwelle im JOG-Betrieb	COLLVLJG	Ganze Zahl 8	Einstellung der Anprechschwelle im JOG-Betrieb (auch während einer Unterbrechung) für jede Achse in % Zur Erhöhung der Empfindlichkeit ist der numerische Wert zu verringern. Spricht die Kollisionsüberwachung im JOG-Betrieb auch ohne einen Zusammenstoß an, erhöhen Sie den Wert. Einstellbereich: 1 bis 500 %	Der Standardwert ist modellabhängig.
Handbedingung	HNDDAT0	Reelle Zahl 7	Einstellung der Handbedingungen beim Start (Festlegung im Werkzeugkoordinatensystem) Nach Einschalten der Spannungsversorgung werden im JOG-Betrieb die hier festgelegten Werte verwendet. Bei Aktivierung der Kollisionsüberwachung im JOG-Betrieb müssen diese Werte eingestellt werden, da die Überwachung sonst ungewollt ansprechen kann. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Der Standardwert ist modellabhängig.
Werkstückbedingung	WRKDAT0	Reelle Zahl 7	Einstellung der Werkstückbedingungen beim Start (Festlegung im Werkzeugkoordinatensystem) Nach Einschalten der Spannungsversorgung werden im JOG-Betrieb die hier festgelegten Werte verwendet. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Nur für die Robotermodelle RV-SD und RH-SDH 0.0,0.0,0.0,0.0,0.0,0.0,0.0

Tab. 3-3: *Parameter für die Kollisionsüberwachung*

① Die Funktion kann nicht in Kombination mit der Funktion zur Steuerung zusätzlicher Mechanismen verwendet werden.

Einstellung der Kollisionsüberwachung

Standardmäßig ist die Ansprechschwelle der Kollisionsüberwachung hoch eingestellt. Zur Erhöhung der Empfindlichkeit ist die Ansprechschwelle über den Parameter COLLVLJG abzusenken. Achten Sie auch auf eine korrekte Einstellung der Parameter HNDDAT0 und WRKDAT0. Werden diese Werte nicht eingestellt, kann die Kollisionsüberwachung in Abhängigkeit der Roboterstellung ungewollt ansprechen.

ACHTUNG:

Die Kollisionsüberwachung schützt den Roboter, den Handgreifer, das Werkstück und andere Komponenten nicht vor Beschädigungen durch einen Zusammenstoß mit umliegenden Einrichtungen. Verfahren Sie den Roboter deshalb immer so, dass es zu keinen Kollisionen kommt.

Betrieb nach einem Zusammenstoß

Um ein Einschalten der Servospannung nach einem Zusammenstoß zu verhindern, erfolgt in diesem Fall nach dem Ansprechen der Kollisionsüberwachung beim Einschalten der Servospannung eine erneute Auslösung der Kollisionsüberwachung. Erfolgt auch weiterhin bei jedem Einschalten der Servospannung eine Fehlerausgabe, lösen Sie die Bremsen des Roboterarms (siehe auch Abschn. 3.19 „Gelenkbremsen lösen“) und schalten Sie dann die Servospannung erneut ein.

Weiterhin ist ein Einschalten der Servospannung auch nach dem temporären Zurücksetzen des Fehlers möglich (siehe auch Abschn. 3.13 „Fehler temporär zurücksetzen“).

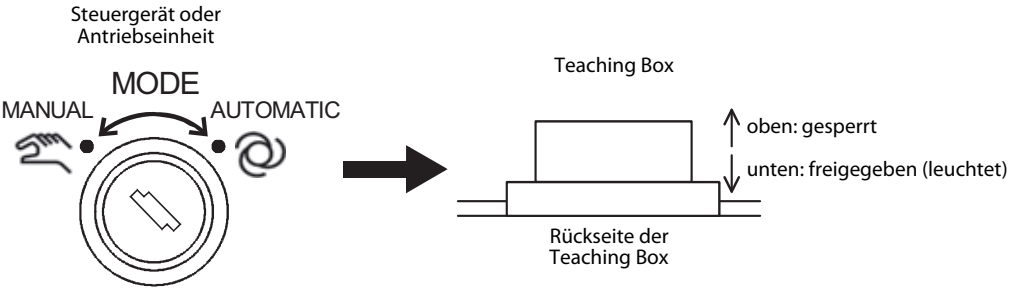
Unterschied zur Kollisionsüberwachung im Automatikbetrieb

Die Einstellung für die Kollisionsüberwachung im JOG-Betrieb ist unabhängig von der Einstellung der Kollisionsüberwachung im Automatikbetrieb. Die Einstellung für den JOG-Betrieb ist wirksam, wenn der Roboter sich nicht im Programmbetrieb befindet. Auch wenn die Kollisionsüberwachung für den Automatikbetrieb deaktiviert und für den JOG-Betrieb aktiviert ist, wird bei einer Unterbrechung des Programmbetriebs die Einstellung für den JOG-Betrieb (Kollisionsüberwachung aktiviert) wirksam.

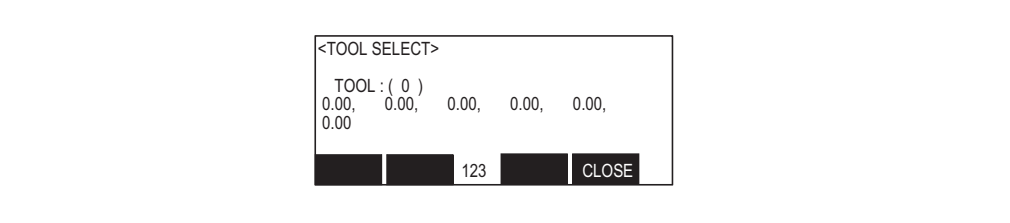
3.5 Werkzeugdaten umschalten

Geben Sie zur Umschaltung der Werkzeugdaten die Werkzeugdaten in die Parameter MEXTL1 bis 16 ein und wählen Sie das Werkzeug wie in folgenden Schritten gezeigt.

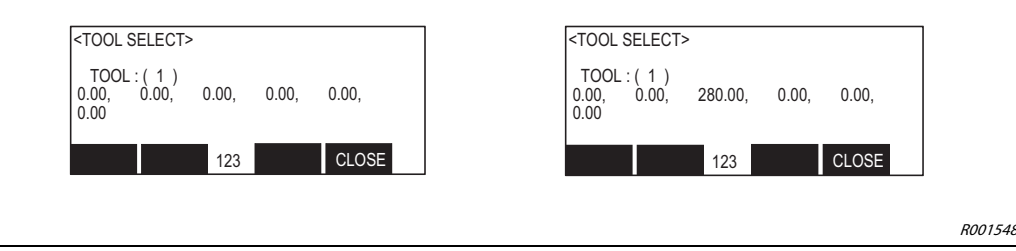
① Stellen Sie den [MODE]-Schalter des Steuergerätes auf die Stellung „MANUAL". Aktivieren Sie die Teaching Box, indem Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „freigegeben" stellen.



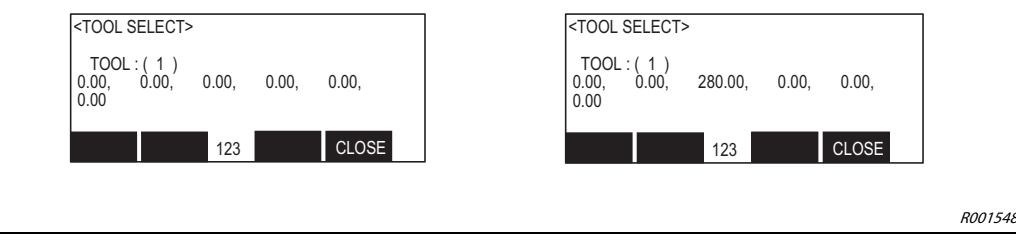
② Betätigen Sie die Taste [HAND] so lange, bis das Werkzeugmenü erscheint.



③ Sie können nun die Daten für die Werkzeuge 1 bis 16 einstellen. Betätigen Sie dazu die entsprechende Zifferntaste und anschließend die [EXE]-Taste. Die Parameter MEXTL1 bis MEXTL16 entsprechen den Ziffern 1 bis 16.



④ Betätigen Sie die entspechende Funktionstaste (F4), um das Menü zu schließen und die Einstellung zu beenden.



⑤ Die aktuellen Werkzeugnummern T1 bis T16 werden oben rechts im JOG-Menü angezeigt.

HINWEISE

Soll der Roboter bei einer Umschaltung der Werkzeugdaten (Parameter MEXTL1 bis 16) im Automatikbetrieb zu der eigentlich geteachten Position bewegt werden, schreiben Sie die entsprechende Werkzeugnummer in die Variable M_TOOL und fahren Sie die Position an, indem Sie die Werkzeugdaten umschalten. Beachten Sie, dass der Roboter unvorhersehbare Bewegungen ausführen kann, falls die Werkzeugdaten beim Teachen nicht mit denen im Automatikbetrieb übereinstimmen.

Wird der Roboter bei einer Umschaltung der Werkzeugdaten während der Ausführung des Programms im Schrittbetrieb bewegt, kann der Roboter unvorhersehbare Bewegungen ausführen, falls die geteachten Werkzeugdaten nicht mit denen der im Schrittbetrieb verwendeten Werkzeugnummer übereinstimmen.

Die aktuelle Werkzeugnummer kann über das Werkzeugmenü, das JOG-Menü oder die Variable M_TOOL angezeigt werden.

Die aktuellen Werkzeugdaten sind im Parameter MEXTL gespeichert. Beachten Sie, dass der Wert bei Auswahl der Werkzeugdaten über die Parameter MEXTL1 bis 16 überschrieben wird. Um die Werkzeugnummer wieder auf „0“ zu setzen, führen Sie den Befehl TOOL aus.

Weitere Informationen zu den Werkzeugdaten finden Sie bei den Parametern MEXTL, MEXTL1, MEXTL2, MEXTL3, MEXTL4 bis MEXTL16, beim Befehl TOOL und bei der Variablen M_TOOL.

3.6 Weltkoordinaten ändern (Festlegung der Nummer des Basiskoordinatensystems)

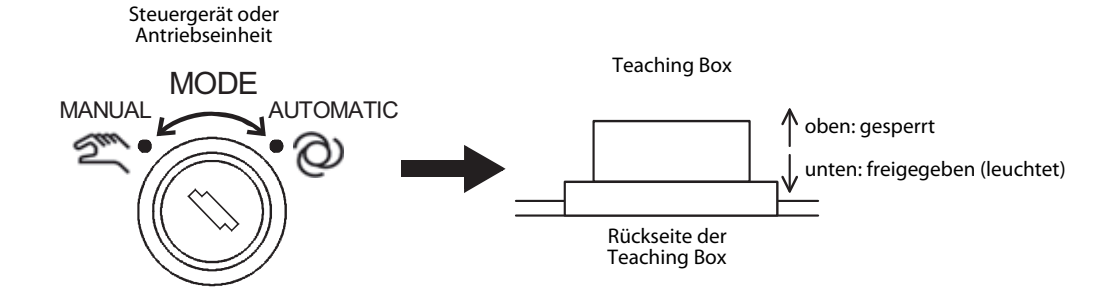
Das Weltkoordinatensystems stellt die Referenz zur Steuerung der aktuellen Roboterposition dar. Es kann auf einfache Art und Weise mit der Teaching Box geändert werden.
Die Verwendung der Basiskonvertierungsfunktion (Base-Befehl) ermöglicht eine komfortableÄnderung der Weltkoordinaten im Teach-Betrieb.

Stellen Sie zuerst die Parameter WK1CORD bis WK8CORD ein. (Eine detaillierte Beschreibung der Parameter WK1CORD bis WK8CORD finden Sie in Abschn. 9.2.)

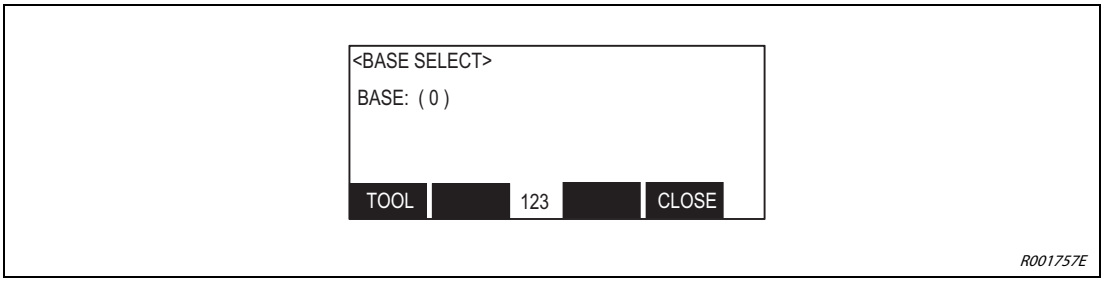
ACHTUNG:

Wird das Weltkoordinatensystem mit dieser Funktion geändert, ändert sich – obwohl der Roboter sich nicht bewegt – auch der aktuelle Koordinatenwert.
Stellen Sie sicher, dass die Beziehung der geteachten Positionen im Programm zum Base-Befehl und zur Nummer des Basiskoordinatensystems korrekt ist.
Das Teachen von Positionen im falschen Basiskoordinatensystem kann bei der Programmausführung zu unvorhersehbaren Bewegungen des Roboters führen. Es besteht Verletzungsgefahr und es können Beschädigungen auftreten.
Benennen Sie die Positionsvariable passend zur Nummer des Basiskoordinatensystem. Führen Sie die Schritte sehr sorgfältig aus.

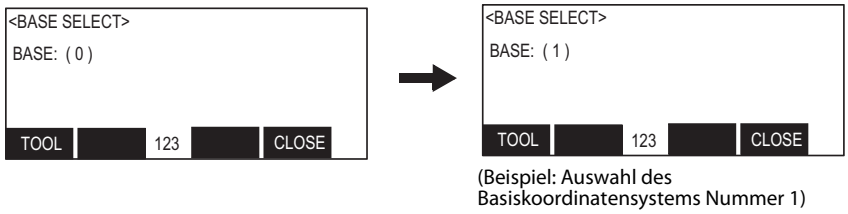
① Stellen Sie den [MODE]-Schalter des Steuergerätes auf die Stellung „MANUAL“. Aktivieren Sie die Teaching Box, indem Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „freigegeben“ stellen.



② Betätigen Sie die Taste [HAND] so lange, bis das Basismenü erscheint. Erscheint das Werkzeugmenü, betätigen Sie die entsprechende Funktionstaste (F1), um das Basismenü aufzurufen.



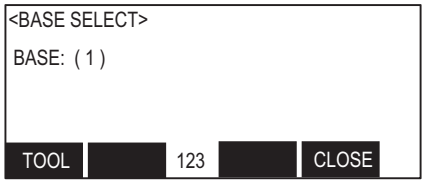
③ Sie können nun die Nummer des Basiskoordinatensystems einstellen. Betätigen Sie dazu die entsprechende Zifferntaste und anschließend die [EXE]-Taste.
1–8: Nummer des Basiskoordinatensystems (entsprechend den Parametern WK1CORD bis WK8CORD
0: Auf Werkseinstellung zurücksetzen (Bedingung ohne Basiskonvertierung)



(Beispiel: Auswahl des Basiskoordinatensystems Nummer 1)

R001758E

④ Betätigen Sie die entspechende Funktionstaste (F4), um das Menü zu schließen und die Einstellung zu beenden.



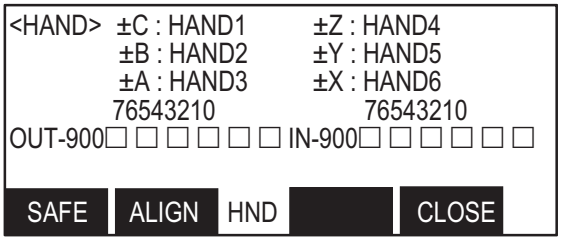
R001548E

⑤ Die aktuelle Nummer des Basiskoordinatensystems B1 bis B8 werden oben rechts im JOG-Menü angezeigt.

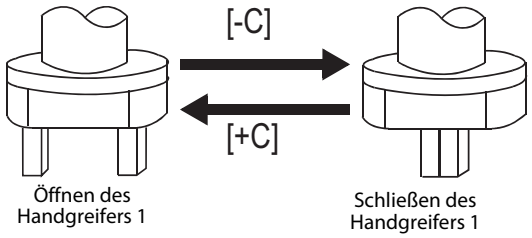
3.7 Handgreifer öffnen/schließen

Die Handgreifer 1 bis 6 können über die Teaching Box geöffnet und geschlossen werden. Gehen Sie zum Öffnen bzw. Schließen eines Handgreifers wie folgt vor:

① Betätigen Sie die [HAND]-Taste, um das HAND-Menü aufzurufen.



② Öffnen und schließen Sie die Hand durch Betätigung der entsprechenden Taste.



Handgreifer	Beschreibung	Tastenbetätigung
1	Öffnen	[+C]
	Schließen	[−C]
2	Öffnen	[+B]
	Schließen	[−B]
3	Öffnen	[+A]
	Schließen	[−A]
4	Öffnen	[+Z]
	Schließen	[−Z]
5	Öffnen	[+Y]
	Schließen	[−Y]
6	Öffnen	[+X]
	Schließen	[−X]

Abb. 3-11: *Öffnen und Schließen eines Handgreifers*

OUT-900 bis OUT-907	7	6	5	4	3	2	1	0
Öffnen/schließen	Schließen	Öffnen	Schließen	Öffnen	Schließen	Öffnen	Schließen	Öffnen
Handnummer	4		3		2		1	

Tab. 3-4: *Handausgänge*

IN-900 bis IN-907	7	6	5	4	3	2	1	0
Eingangssignal	907	906	905	904	903	902	901	900

Tab. 3-5: *Handeingänge*

Im Handbereich des Roboters können unterschiedliche Werkzeuge montiert werden. Erfolgt bei einer pneumatischen Greifhand die Ansteuerung über ein Zweifach-Ventil, dienen zwei Bits des Handsteuersignals der Überwachung des Handgreiferzustands. Weitere Informationen über die Handsteuersignale finden Sie in Abschn. 9.13 und Abschn. 9.14.

3.8 Handgreifer ausrichten

Ein Ausrichten des Handgreifers bewirkt eine Bewegung der Hand zu der Position, die den kleinstmöglichen Weg zur senkrechten oder waagerechten Stellung der Achsen A, B und C hat.

Sind die Werkzeugkoordinaten über den Tool-Befehl oder über Parameter definiert, erfolgt die Ausrichtung der Hand in den festgelegten Koordinaten. Sind die Koordinaten nicht definiert, erfolgt die Ausrichtung der Hand im Mittelpunkt des Handflansches. Eine detaillierte Beschreibung der Werkzeugkoordinaten finden Sie in Abschn. 9.7.

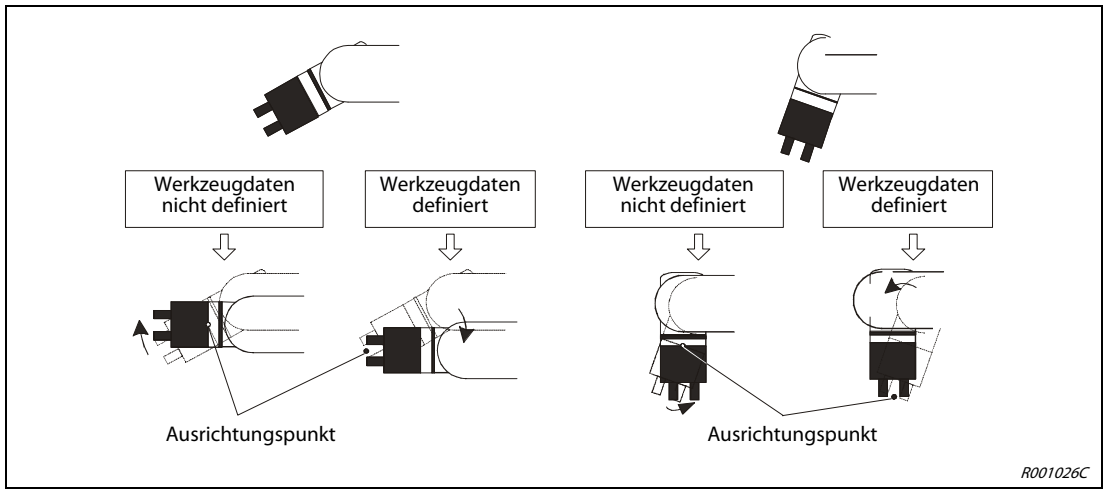
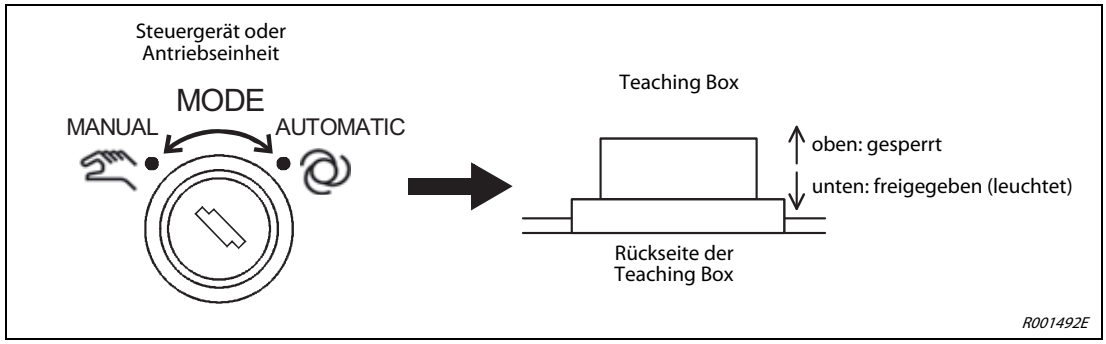


Abb. 3-12: *Ausrichten des Handgreifers*

Gehen Sie zum Öffnen bzw. Schließen eines Handgreifers wie folgt vor:

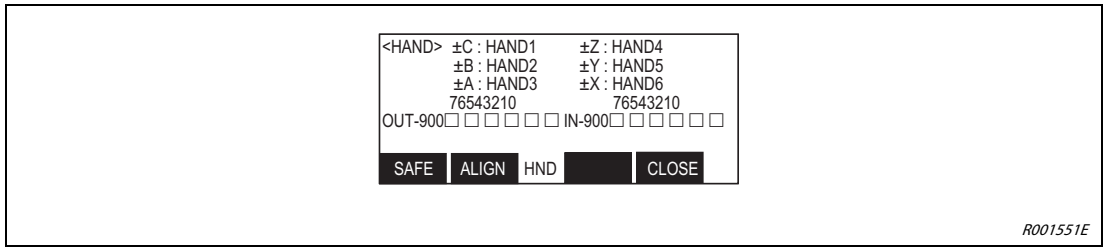
① Aktivieren Sie die Teaching Box, indem Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „freigegeben“ stellen.



② Halten Sie den Dreistufenschalter in Mittelstellung und betätigen Sie die [Servo]-Taste, um die Servospannung einzuschalten.

③ Betätigen Sie die [HAND]-Taste, um das HAND-Menü aufzurufen.

④ Halten Sie den Dreistufenschalter in Mittelstellung und betätigen Sie die Funktionstaste, die der Funktion „Alignment“ zugeordnet ist. Die Hand wird während der Tastenbetätigung ausgerichtet und die START-LED des Steuergeräts leuchtet.



HINWEIS

Nimmt eines der Orientierungsdatenelemente A, B oder C nach der Ausrichtung den Wert 180° an, so kann dieser Wert bei gleicher Stellung +180° oder −180° entsprechen. Die Uneindeutigkeit des Vorzeichens kann zu internen Verarbeitungsfehlern führen. Wird eine solche Position z.B. bei der Palettierung verwendet (Def Plt), so kann die Hand rotieren und sich in unvorhersehbarer Weise bewegen, da die Palettierungsfunktion ihre Positionen durch eine Unterteilung der Distanz zwischen −180° und +180° berechnet. Möchten Sie für die Palettierungsfunktion bestimmte Positiondaten verwenden, deren Orientierungsdaten 180° betragen, so versehen Sie diese mit dem gleichen Vorzeichen.
Werden diese Positionsdaten direkt mit einem Interpolationsbefehl verwendet, bewegt sich die Hand unabhängig vom Vorzeichen ohne Schwierigkeiten.

3.9 Programmierung

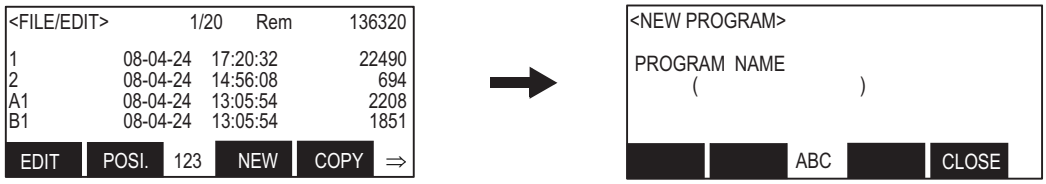
Die Programmiersprache MELFA-BASIC V ermöglicht die Erstellung komplexer Programme mit umfangreichen Funktionen. In diesem Abschnitt wird die Vorgehensweise bei der Programmierung über die Teaching Box erläutert.
Eine detaillierte Beschreibung der MELFA-BASIC-V-Befehle finden Sie in Abschn. 6.3.

3.9.1 Roboterprogramm erstellen

Aufruf des Menüs zur Programmeditierung

① Wählen Sie im Hauptmenü das Menü „FILE/EDIT“ aus.

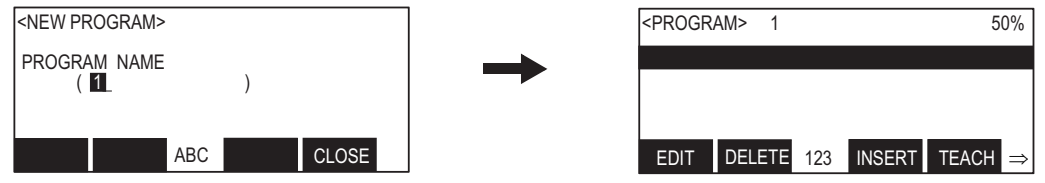
② Betätigen Sie die Funktionstaste, die der Funktion „NEW“ zugeordnet ist.



Betätigen Sie die Funktionstaste F3.

R001553E

③ Geben Sie den Progammnamen ein. Rufen Sie das Menü zur Programmeditierung auf. (Wurde der Name eines bereits vorhandenen Programms eingegeben, wird dieses Proramm aufgerufen.)



Die Programmnummer „1“ wird ausgewählt.
Betätigen Sie dazu die Tasten [1][EXE].

R001554E

Erstellung eines neuen Programms

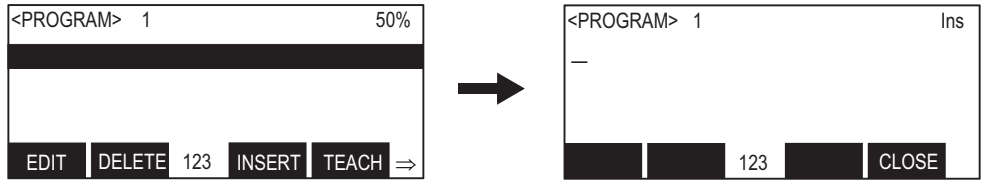
Folgendes Programm soll erstellt werden:

1 Mov P1

2 Mov P2

3 End

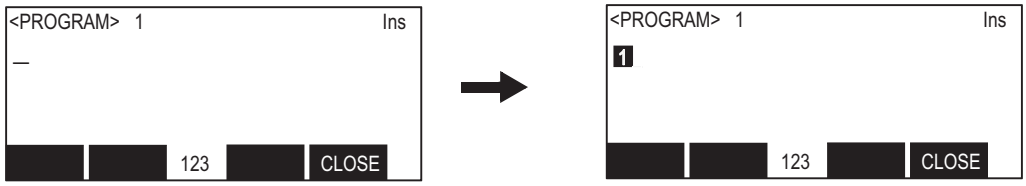
① Betätigen Sie die Funktionstaste ([F3]), die im Menü zur Programmeditierung der Funktion „INSERT“ zugeordnet ist.



Betätigen Sie die Taste [F3],
um einen Programmschritt einzufügen.

R001555E

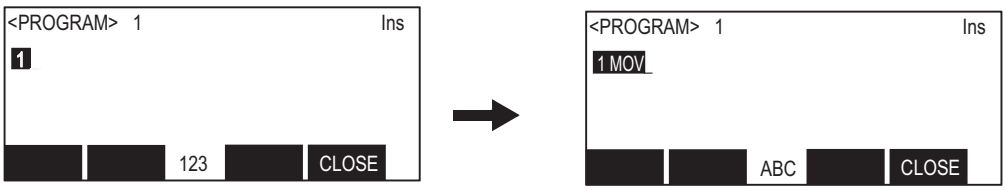
② Eingabe des Programmschritts „1“
Ist der Eingabemodus für Zahlen noch nicht ausgewählt, betätigen Sie die [CHARACTER]-Taste, um in diesen Modus zu wechseln. Betätigen Sie die Taste [1]. Die Leerstelle zwischen Schrittnummer und Befehl kann weggelassen werden.



Betätigen Sie die Taste [1],
um die Schrittnummer einzugeben.

R001556E

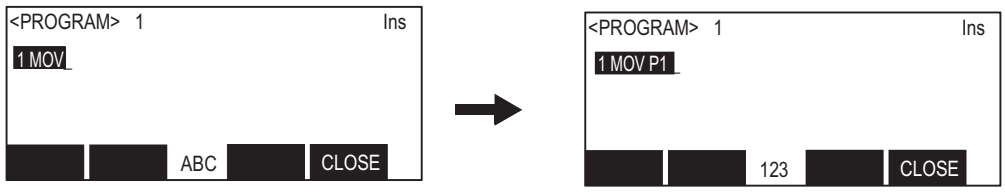
③ Eingabe des Befehls „Mov“
Betätigen Sie die [CHARACTER]-Taste, um in den Eingabemodus für Buchstaben zu wechseln. Betätigen Sie die Tasten 1 × [MNO], [→], 3 × [MNO], 3 × [TUV].



Betätigen Sie die Tasten [CHARACTER], 1 × [MNO], [→], 3 × [MNO], 3 × [TUV]

R001557E

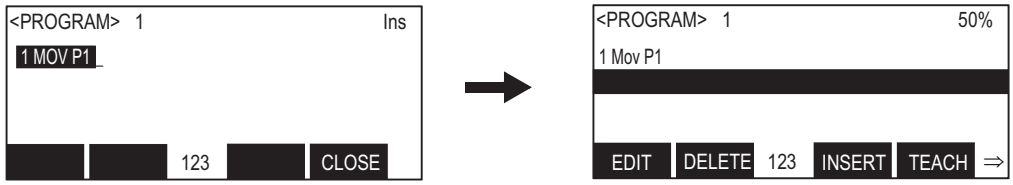
④ Eingabe der Positionsvariablen „P1“
Betätigen Sie die Tasten [SP] (Leerzeichen) und [PQRS].
Betätigen Sie die [CHARACTER]-Taste, um in den Eingabemodus für Zahlen zu wechseln.
Geben Sie die Ziffer „1“ ein.



Betätigen Sie die Tasten [SP], [PQRS], [CHARACTER] und [1], um ein Leerzeichen und die Zeichen „P1“ einzugeben.

R001558E

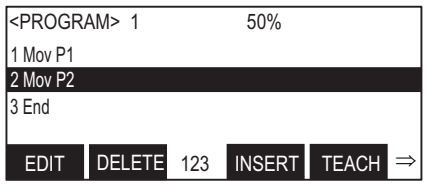
⑤ Speichern des ersten Programmschritts
Betätigen Sie die Taste [EXE], um den ersten Programmschritt zu speichern.



Betätigen Sie die Taste [EXE], um den ersten Programmschritt zu speichern.

R001559E

⑥ Geben Sie die Programmschritte 2 und 3 in derselben Weise ein.



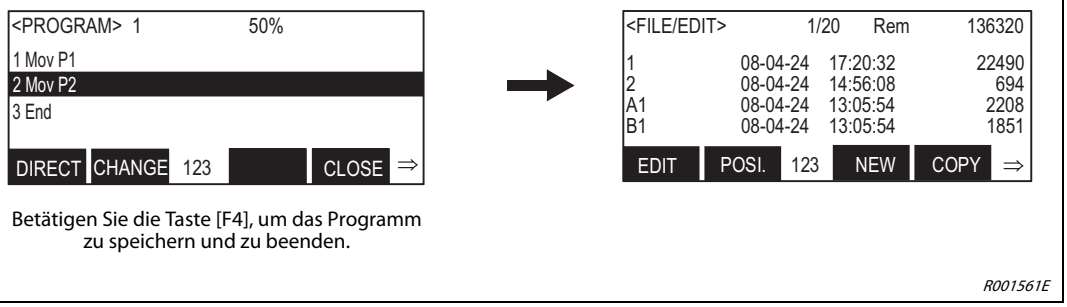
R001560E

Beschreibung:

- Anzeige der vorhergehenden/nächsten Programmzeile
  Auf der Teaching Box werden vier Programmzeilen angezeigt. Die vorhergehende Programmzeile wird durch Betätigung der [↑]-Taste, die nächste Programmzeile durch Betätigung der [↓]-Taste aufgerufen.
- Anzeige eines bestimmten Programmschritts
  Betätigen Sie die [FUNCTION]-Taste, um die Funktionsanzeige zu ändern, und betätigen Sie die Taste [F2]. Das JUMP-Menü wird aufgerufen. Geben Sie die Nummer des Schritts, den Sie aufrufen möchten, ein und betätigen Sie die [EXE]-Taste. Der aufgerufene Programmschritt erscheint.
- Schrittnummer
  Die Schrittnummer kann bei der Eingabe weggelassen werden. Sie wird beim Aufruf des nächsten Schritts automatisch eingefügt.
- Groß-/Kleinbuchstaben
  Reservierte Wörter und Variablennamen werden in MELFA-BASIC V durch eine Kombination von Groß- und Kleinbuchstaben dargestellt. Die Umschaltung von Groß- auf Kleinbuchstaben erfolgt bei Eingabe über die Teaching Box automatisch mit der Bestätigung eines Programmschritts.

Programm speichern

Bei Betätigung der Funktionstaste, der die Funktion „CLOSE“ zugeordnet ist, wird die Programmeingabe abgeschlossen und das Programm gespeichert. Wird die Funktion „CLOSE“ nicht angezeigt, betätigen Sie die [FUNCTION]-Taste.



Betätigen Sie die Taste [F4], um das Programm zu speichern und zu beenden.

HINWEISE

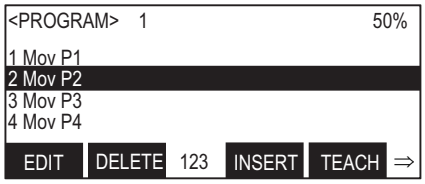
Wird bei angezeigtem Programmeingabe-Bildschirm die Spannungsversorgung abgeschaltet, werden die eingegebenen Programmabschnitte gelöscht.

Speichern Sie aus Sicherheitsgründen Programme nicht nur im Steuergerät, sondern erstellen Sie auch Sicherheitskopien auf einem PC. Verwenden Sie dazu die optionale Software RT ToolBox2.

3.9.2 Roboterprogramm editieren

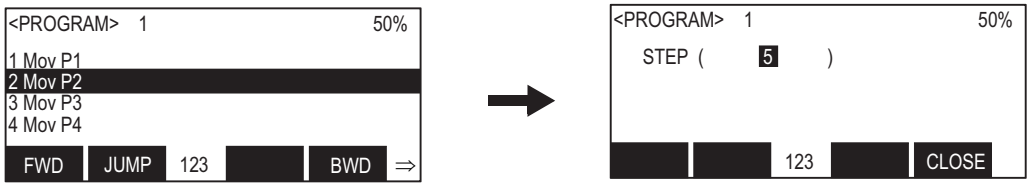
Rufen Sie das Menü zur Programmeditierung auf (siehe Abschn. 3.9.1). Es wird automatisch das Programmverzeichnis angezeigt, wenn Sie keine Programmauswahl vornehmen.

Beispiel▽ Die Zeile 5 in folgendem Programm soll von „5 Mov P5“ auf „5 Mvs P5“ geändert werden.



R001562E

① Anzeige des Programmschritts 5
Betätigen Sie die [FUNCTION]-Taste, um die Funktionsanzeige umzuschalten. Betätigen Sie die Taste [F2] (JUMP), um das Menü zur Eingabe der Schrittnummer aufzurufen. Betätigen Sie die Tasten [5] und [EXE], um den fünften Programmschritt aufzurufen.
Sie können den fünften Programmschritt auch über die Tasten [↑] und [↓] aufrufen.

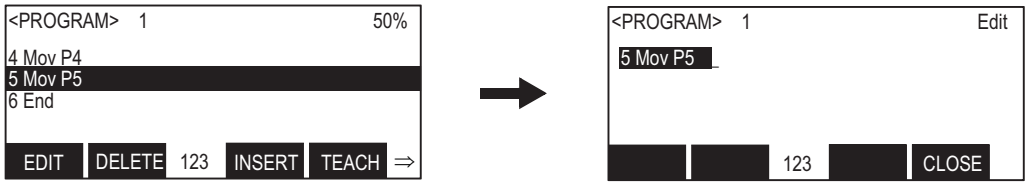


Betätigen Sie die Taste [F2], um das Menü zur Befehlseingabe aufzurufen.

Betätigen Sie die Taste [5] und [EXE], um den fünften Programmschritt aufzurufen.

R001563E

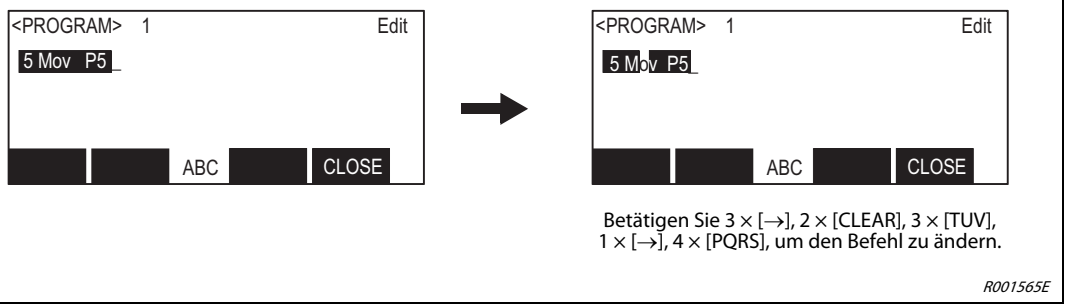
② Editieren des Programmschritts 5
Betätigen Sie die Funktionstaste, die der Funktion „Edit“ zugeordnet ist.



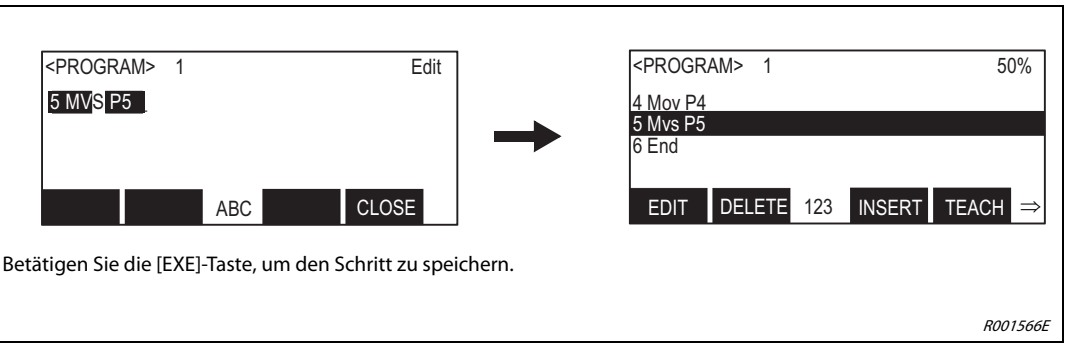
Betätigen Sie die Taste [F1], um den fünften Programmschritt zu editieren.

R001564E

③ Editieren des Befehls
Betätigen Sie 3 × [→], um den Cursor zum Zeichen „o“ zu bewegen.
Betätigen Sie 2 × [CLEAR], um die Zeichen „ov“ zu löschen. Das Zeichen „M“ bleibt stehen. Betätigen Sie 3 × [TUV], [→], 4 × [PQRS], um die Zeichen „vs“ einzugeben. Der Befehl in Schritt 5 lautet dann „Mvs P5“.



④ Betätigen Sie die [EXE]-Taste, um den Schritt zu speichern.



△

Beschreibung

- Auswahl des Programmschritts
  Sie können den fünften Programmschritt auch über die Tasten [↑] und [↓] aufrufen und die Zeile dann nach Betätigung der Funktionstaste, die der Funktion „Edit“ zugeordnet ist, bearbeiten.
- Abbruch der Editierung
  Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um die Editierung abzubrechen.
- Editieren eines Zeichens
  Betätigen Sie zum Löschen eines Zeichens die Taste [CLEAR], nachdem Sie den Cursor auf dem zu löschenden Zeichen platziert haben, und geben Sie das korrekte Zeichen ein.
- Prüfen des editierten Roboterprogramms
  Speichern Sie das Roboterprogramm nach erfolgter Editierung durch Betätigung der Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist [F4] oder betätigen Sie den [ENABLE/DISABLE]-Schalter auf der Rückseite der Teaching Box, um die Teaching Box zu deaktivieren. Prüfen Sie anschließend die ausgeführten Korrekturen im Schrittbetrieb.

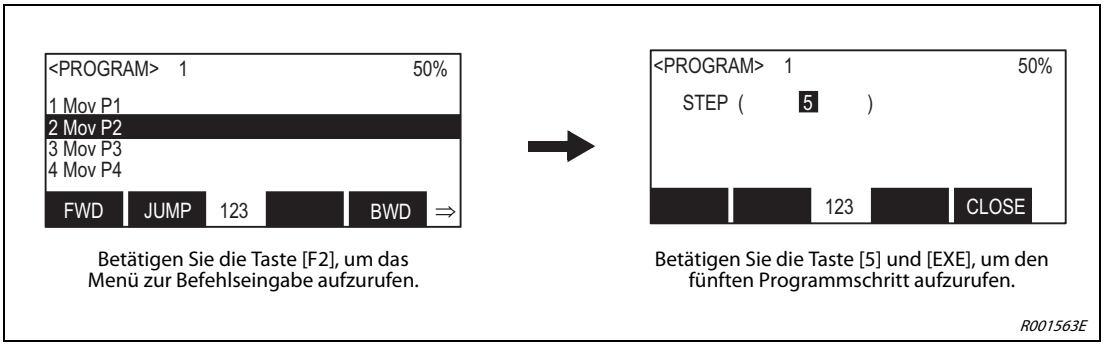
Eingabe der aktuellen Positionsdaten

Positionen, die der Roboter im JOG-Betrieb o. Ä. anfährt, und die im Programm verwendet werden sollen, können geteacht werden. Das erneute Teachen einer Position überschreibt die alten Werte. Das Teachen von Positionen kann im Menü zur Programmeditierung und im Menü zur Positionseditierung erfolgen.

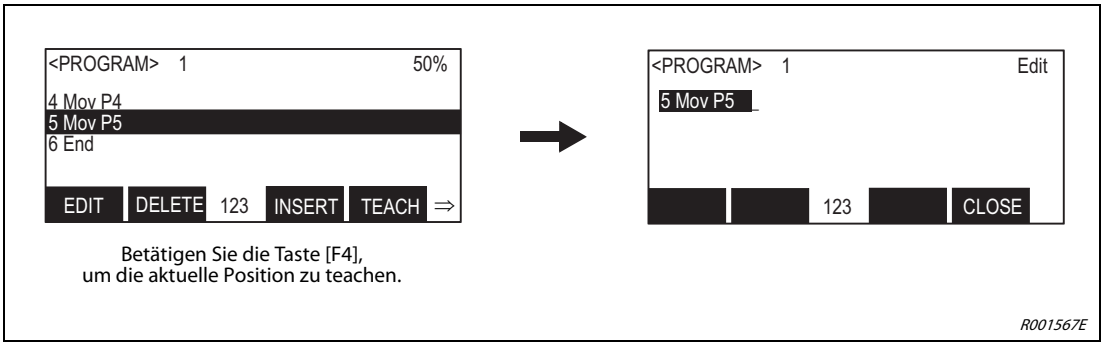
- Teachen im Menü zur Programmeditierung

Beispiel▽ Rufen Sie den Programmschritt auf, in dem die zu teachende Position verwendet wird. In diesem Beispiel soll die Positionsvariable P5 im Programmschritt 5 „Mvs P5" geteacht werden. Fahren Sie zuerst die Position P5 im JOG-Betrieb o. Ä. an.

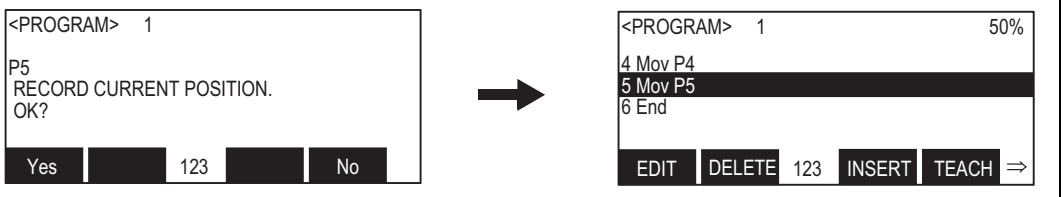
① Anzeige des Programmschritts 5
Betätigen Sie die Taste [F2] (JUMP), um das Menü zur Eingabe der Schrittnummer aufzurufen.
Betätigen Sie die Tasten [5] und [EXE], um den fünften Programmschritt aufzurufen.
Sie können den fünften Programmschritt auch über die Tasten [↑] und [↓] aufrufen.


<PROGRAM> 1 50%
1 Mov P1
2 Mov P2
3 Mov P3
4 Mov P4
FWD JUMP 123 BWD ⇒
<PROGRAM> 1 50%
STEP (5)
123 CLOSE
Betätigen Sie die Taste [F2], um das Menü zur Befehlseingabe aufzurufen.
Betätigen Sie die Taste [5] und [EXE], um den fünften Programmschritt aufzurufen.
R001563E

② Teachen der aktuellen Position
Betätigen Sie die Funktionstaste, die der Funktion „TEACH" zugeordnet ist.



③ Betätigen Sie die Funktionstaste, die der Funktion „Yes“ zugeordnet ist, um die Daten der aktuellen Position in die Positionsvariable P5 zu übertragen. Anschließend wird das Menü zur Programmeditierung angezeigt. Möchten Sie den Teach-Vorgang abbrechen, betätigen Sie die Funktionstaste, die der Funktion „No“ zugeordnet ist.



Betätigen Sie die Taste [F1], um die Daten der aktuellen Position in die Positionsvariable P5 zu übertragen.

R001568E

△

HINWEIS

Nur eine Positionsvariable als Ziel

Enthält der Schritt zum Einlesen der Position zwei oder mehr Positionsvariablen, wie z.B. „Mov P1+P2“ oder „P1=P10“, erfolgt die Übertragung der Daten in die Variable, die am weitesten links steht.

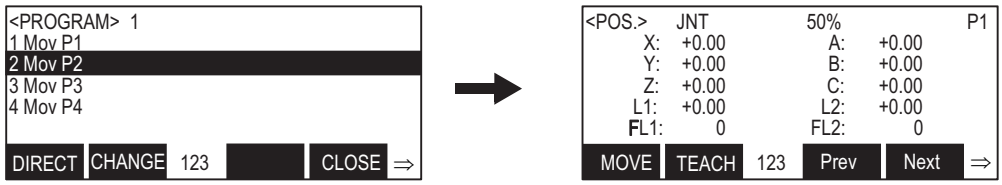
In Fällen, in denen die Groß- und Kleinschreibung der Variablen gemischt ist, wie z.B. „Mov p1+P2“, erfolgt die Übertragung der Daten in die aus Großbuchstaben bestehende Variable. (Das gilt für die Software der Teaching Box ab Version 1.3.)

Andere Variablen werden nach der Methode, die auf der nächsten Seite beschrieben wird, geteacht.

● Teachen im Menü zur Positionseditierung

Beispiel▽ In diesem Beispiel soll die Positionsvariable P5 geteacht werden. Fahren Sie zuerst die Position P5 im JOG-Betrieb o.Ä. an.

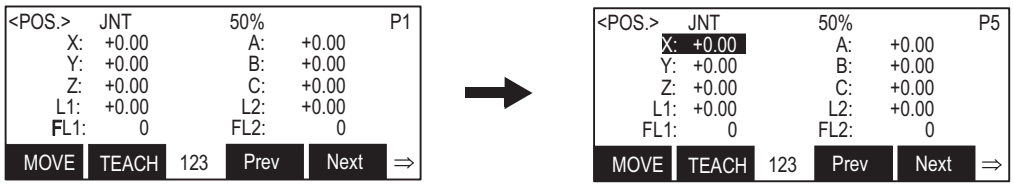
① Betätigen Sie die Taste [F2] (CHANGE), um das Menü zur Positionseditierung aufzurufen.



Betätigen Sie die Taste [F2], um das Menü zur Positionseditierung aufzurufen.

R001569E

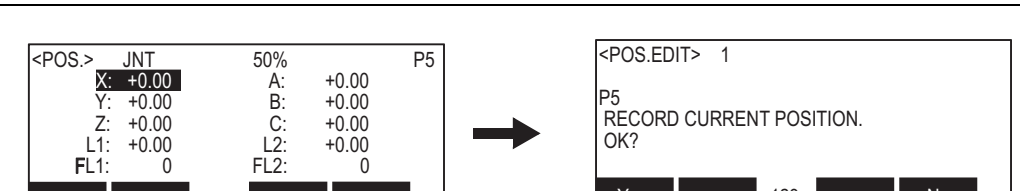
② Betätigen Sie die Funktionstaste die der Funktion „Prev" oder „Next" zugeordnet ist, um die Position P5 aufzurufen.



Betätigen Sie die Taste [F3] oder [F4], um die Position P5 aufzurufen.

R001570E

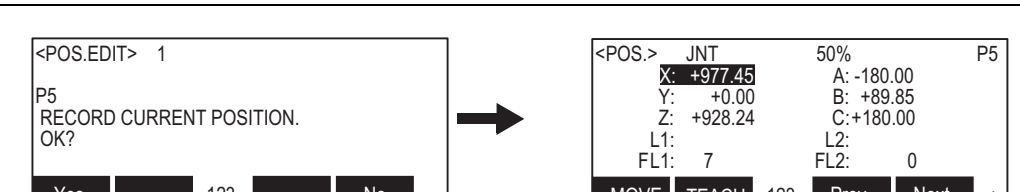
③ Teachen der aktuellen Position
Betätigen Sie die Funktionstaste, die der Funktion „TEACH" zugeordnet ist. Das Bestätigungsmenü erscheint.



Betätigen Sie die Taste [F2], um die aktuelle Position zu teachen.

R001571E

④ Betätigen Sie die Funktionstaste, die der Funktion „Yes" zugeordnet ist, um die Daten der aktuellen Position in die Positionsvariable P5 zu übertragen. Anschließend wird das Menü zur Positionseditierung angezeigt. Möchten Sie den Teach-Vorgang abbrechen, betätigen Sie die Funktionstaste, die der Funktion „No" zugeordnet ist.



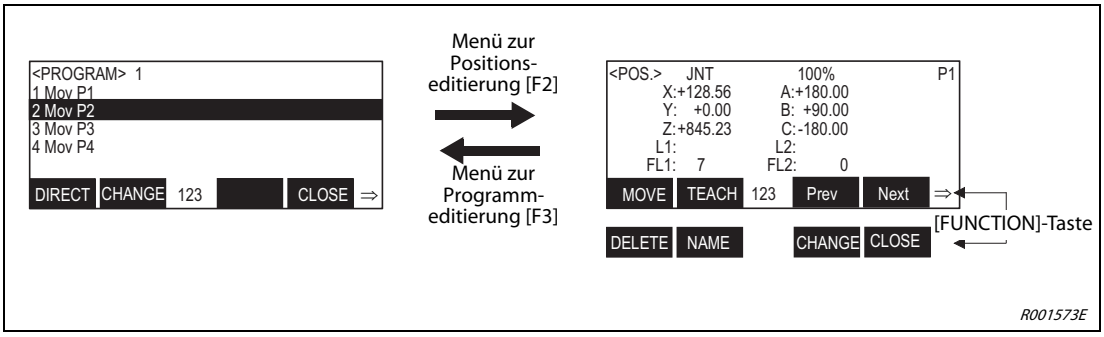
Betätigen Sie die Taste [F1], um die Daten der aktuellen Position in die Positionsvariable P5 zu übertragen.

R001572E

△

Beschreibung

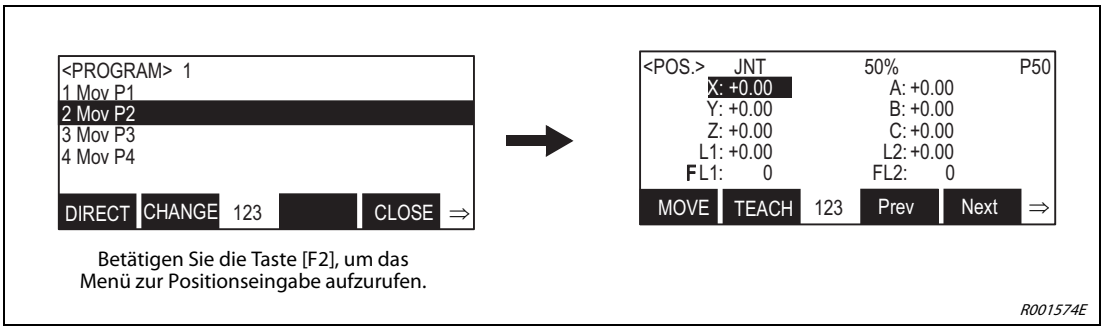
Wechsel zwischen dem Menü zur Programmeditierung und dem Menü zur Positionseditierung
Eine Umschaltung zwischen dem Menü zur Programmeditierung und dem Menü zur Positonseditierung erfolgt durch Betätigung der Funktionstaste, die der Funktion „CHANGE“ zugeordnet ist. Steht die Funktion „CHANGE“ nicht zur Verfügung, wird angezeigt, dass die [FUNCTION]-Taste betätigt ist. Ist das Zeichen „=>“ in der rechten Ecke der Anzeige sichtbar, bedeutet dies, dass ein Wechsel der Anzeige über die Betätigung der [FUNCTION]-Taste erfolgen kann.



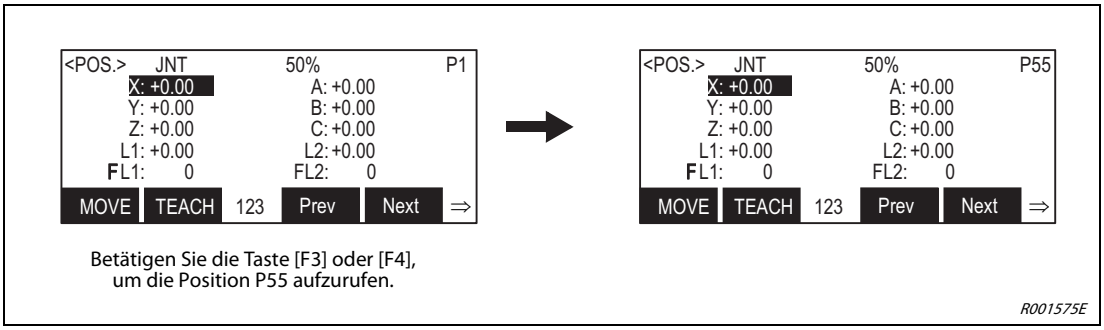
Über die Funktionen „Prev“ [F3] und „Next“ [F4] können die Positionsvariablen der Reihe nach aufgerufen werden. Neben dem Aufruf der Positionsvariablen ist durch Betätigung der Tasten [FUNCTION] und [NAME] auch ein Aufruf der Gelenkvariablen möglich. Auch die Gelenkvariablen können über die Funktionen „Prev“ [F3] und „Next“ [F4] der Reihe nach aufgerufen werden. Bei einem Aufruf der Gelenkvariablen (J), die vor der ersten oder nach der letzten steht, erfolgt eine Umschaltung auf die Positionsvariablen (P).

Positionsdaten löschen

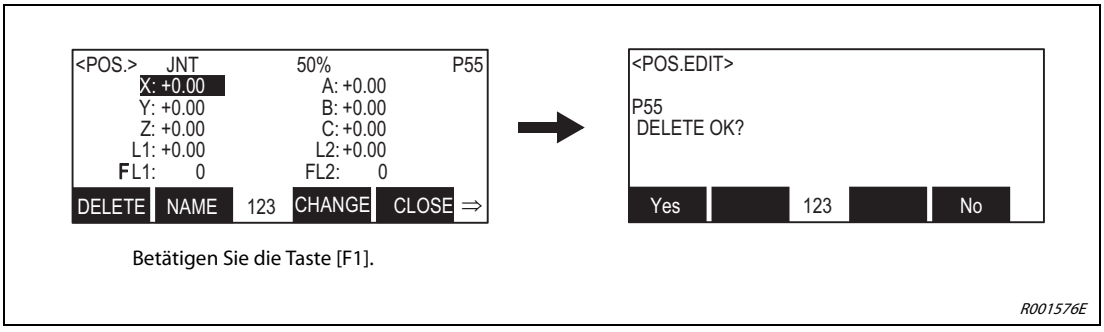
① Aufruf des Menüs zur Positionseditierung
Betätigen Sie die Funktionstaste, die der Funktion „CHANGE“ zugeordnet ist, um das Menü zur Positionseditierung aufzurufen.



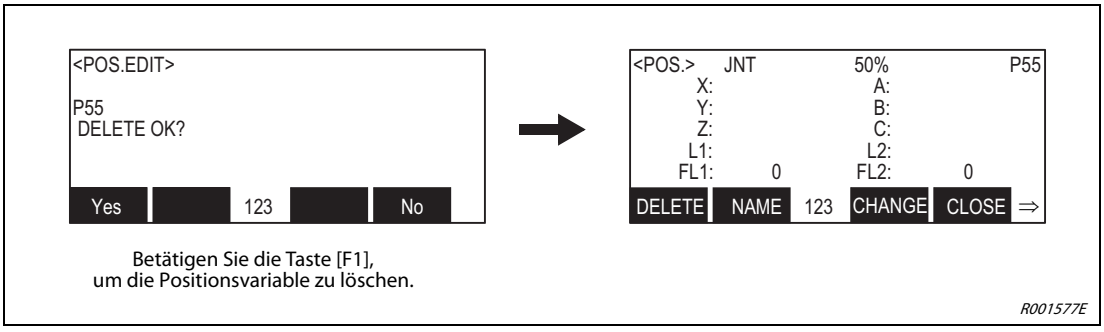
② Anzeige der Variablen, die gelöscht werden soll
Betätigen Sie die Funktionstaste, die der Funktion „Prev“ oder „Next“ zuzgeordnet ist, um die Variable, die gelöscht werden soll, aufzurufen.


<POS.> JNT 50% P1
X: +0.00 A: +0.00
Y: +0.00 B: +0.00
Z: +0.00 C: +0.00
L1: +0.00 L2: +0.00
FL1: 0 FL2: 0
MOVE TEACH 123 Prev Next ⇒
<POS.> JNT 50% P55
X: +0.00 A: +0.00
Y: +0.00 B: +0.00
Z: +0.00 C: +0.00
L1: +0.00 L2: +0.00
FL1: 0 FL2: 0
MOVE TEACH 123 Prev Next ⇒
Betätigen Sie die Taste [F3] oder [F4], um die Position P55 aufzurufen.
R001575E

③ Löschen der Positionsvariablen
Betätigen Sie die Funktionstaste die der Funktion „DELETE“ zugeordnet ist. Das Bestätigungsmenü erscheint. (Wird die Funktion „DELETE“ nicht angezeigt, betätigen Sie die [FUNCTION]-Taste.)



④ Löschen der Positionsvariablen
Betätigen Sie die Funktionstaste die der Funktion „Yes“ zugeordnet ist, um die Positionsvariable zu löschen.



Position anfahren

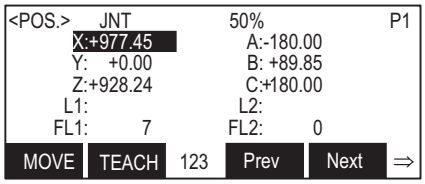
Die Handspitze (TCP: Tool Center Point) des Roboters kann zu einer bereits definierten Position gefahren werden. Das Anfahren der Position erfolgt im Gelenk- oder XYZ-Modus.

Schalten Sie die Servospannung ein, bevor Sie einen Bewegungsbefehl ausführen. Betätigen Sie den Dreistufenschalter, während Sie die Servospannung einschalten. Bei nicht betätigtem Dreistufenschalter lässt sich die Servospannung nicht einschalten.

Bezeichnung	Bewegung
Gelenk-Modus	Die definierte Position wird mittels Gelenk-Interpolation angefahren. Der Bewegungsbefehl wird im Gelenk-JOG-Betrieb verwendet. Die Achsen werden wie beim Mov-Befehl verfahren.
XYZ-Modus	Die definierte Position wird mittels Linear-Interpolation angefahren. Es wird keine Verfahrbewegung ausgeführt, wenn der Stellungsmerker der aktuellen Position von der anzufahrenden Position abweicht. Der Bewegungsbefehl wird im XYZ-, 3-Achsen-XYZ-, Kreis- oder im Werkzeug-JOG-Betrieb verwendet. Die Achsen werden wie beim Mvs-Befehl verfahren.

Tab. 3-6: *Anfahren einer definierten Position*

① Rufen Sie die Position auf, die angefahren werden soll.
Betätigen Sie die Funktionstaste, die der Funktion „MOVE“ zugeordnet ist. Während Sie die Taste betätigen, bewegt der Roboter sich zu der gewünschten Position.
Lassen Sie die Taste los, stoppt der Roboter. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung und der Roboter stoppt ebenfalls.



R001578E

ACHTUNG:

Stellen Sie vor dem Anfahren der Position sicher, dass der Roboter während der Verfahrbewegung nicht mit umliegenden Einrichtungen kollidieren kann.
Führen Sie die Verfahrbewegung zuerst mit einer niedrigen Geschwindigkeit aus. Beachten Sie den unterschiedlichen Verlauf des Verfahrweges in Abhängigkeit des Interpolationsmodus (Gelenk-/XYZ-Modus).

Positionsdaten ändern (manuelle Dateneingabe)

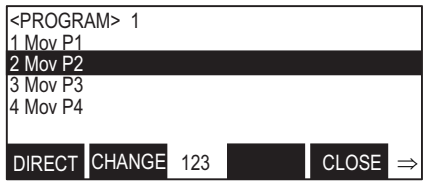
Bei der manuellen Dateneingabe werden die Positionsdaten durch die direkte Eingabe der numerischen Daten für die Achsenelemente geändert.
Die Methode ist gut zur Feinausrichtung einer Position geeignet, da die Positionsdaten den relativen Abstand zu einem Bezugspunkt angeben.

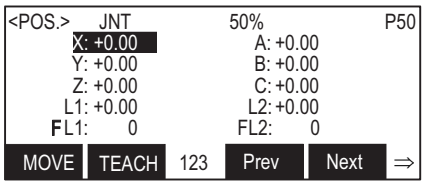
Beispiel▽ Eine Position, die 50 mm über der Position P1 liegt, soll mittels Gelenk-Interpolation angefahren werden. Die Position P1 wurde dabei geteacht. Setzen Sie die Z-Achse der Position P1 nun auf „0.00“ und die Z-Achse der Position P50 auf „50.00“, kann die gewünschte Position mit dem Befehl Mov P1+P50 angefahren werden.

△

Folgende Schritte zeigen die Einstellung der Daten für die Position P50 aus dem obigen Beispiel

① Aufruf des Menüs zur Positionseditierung
Betätigen Sie die Funktionstaste, die der Funktion „CHANGE“ zugeordnet ist, um das Menü zur Positionseditierung aufzurufen.

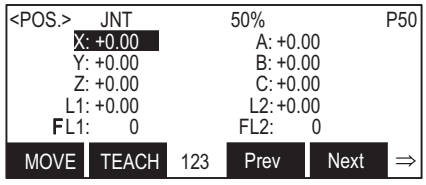


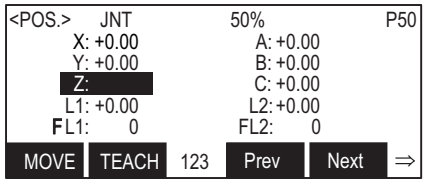


Betätigen Sie die Taste [F2], um das Menü zur Positionseingabe aufzurufen.

R001574E

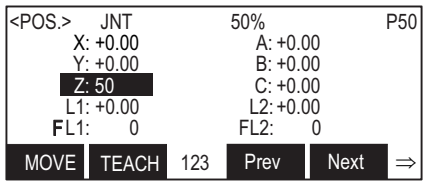
② Eingabe des Werts „50,00“ für die Z-Achse
Betätigen Sie 2 × [↓], um den Cursor zur Z-Achse zu bewegen. Betätigen Sie die Taste [CLEAR], um den aktuell angezeigten Wert „0.00“ zu löschen. Betätigen Sie die Tasten [5], [0] und [EXE], um den Wert der Z-Achse auf 50 mm zu setzen.

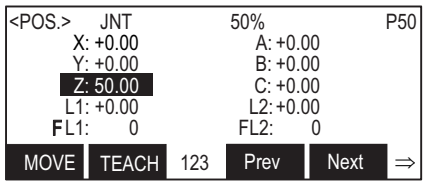




Betätigen Sie die Tasten [↓] und [CLEAR], um den aktuell angezeigten Wert zu löschen.

R001579E





Betätigen Sie die Tasten [5], [0] und [EXE], um den Wert für die Z-Achse auf 50 mm zu setzen.

R001580E

3.9.3 Roboterprogramm testen

Nach der Programmerstellung sollten Sie das Programm „testen“. Mit Testen ist die Suche nach und die Beseitigung von Programmfehlern gemeint.

Das Testen des Programms erfolgt mit der Teaching Box.

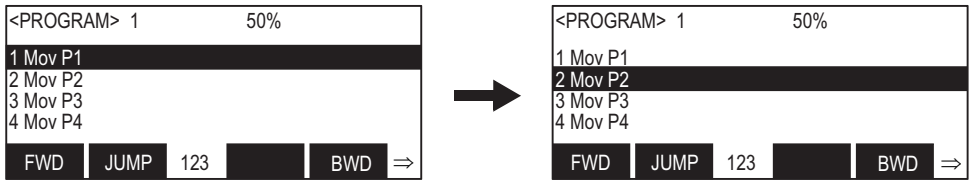
ACHTUNG:

Testen Sie unbedingt jedes Roboterprogramm vor einem automatischen Betriebseinsatz!

Programm schrittweise ausführen (vorwärts)

Die Programmausführung erfolgt zeilenweise in Vorwärtsrichtung. Rufen Sie das Menü zur Programmeditierung auf, um das Programm zu testen. Betätigen Sie nach Einschalten der Servospannung bei der Ausführung der folgenden Schritte den Dreistufenschalter auf der Rückseite der Teaching Box.

① Betätigen Sie die [FUNCTION]-Taste, um die Funktionsanzeige zu ändern. Betätigen Sie die Taste [F1](FWD), um die Roboterbewegung zu starten. Nach Abarbeitung eines Schritts stoppt der Roboter und der nächste Schritt wird angezeigt.
Wird die [F1]-Taste während der Bewegung losgelassen , stoppt der Roboter. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung und der Roboter stoppt ebenfalls.
Während der Ausführung eines Programmschritts leuchtet die LED der [START]-Taste am Steuergerät. Nach der Ausführung eines Schritts erlischt die LED und die LED der [STOP]-Taste leuchtet. Wird danach die [F1]-Taste losgelassen, springt der Cursor zum nächten Schritt.
Bei jeder Betätigung der [F1]-Taste wird der nächste Schritt ausgeführt.



Betätigen Sie die Taste [F1],
um den Schrittbetrieb zu starten.

R001581E

ACHTUNG:

- ***Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.***
- ***Der genaue Verfahrweg des Roboters kann sich in Abhängigkeit der Geschwindigkeit ändern. Besonders im Bereich der Eckpunkte kann der Verfahrweg variieren. Verfahren Sie den Roboter beim Start des Automatikbetriebs daher erst mit niedriger Geschwindigkeit. Steigern Sie die Geschwindigkeit allmählich und achten Sie darauf, dass keine Kollisionen mit umliegenden Einheiten auftreten.***

Beschreibung

- Schrittbetrieb
  Im Schrittbetrieb wird das Roboterprogramm zeilenweise abgearbeitet. Die Verfahrgeschwindigkeit ist niedrig und der Roboter stoppt nach Abarbeitung jeder Zeile, um eine Überprüfung der Programmfunktionen und der Verfahrbewegung zu ermöglichen. Während des Schrittbetriebs leuchtet die grüne LED des [START]-Tasters am Steuergerät.
  Bei Ausführung eines End- oder Hlt-Befehls wird der Schrittbetrieb abgebrochen.

- Aufruf eines anderen Schritts
  Mit Hilfe der Pfeiltasten oder der Funktion „Jump“ können Sie einen anderen Schritt aufrufen.

- Stoppen des Roboters im Betrieb

  – durch Betätigung des NOT-HALT-Schalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Setzen Sie zur Überprüfung des Betriebs den Fehler zurück, schalten Sie die Servoversorgung wieder ein und führen Sie das Programm im Schrittbetrieb aus.

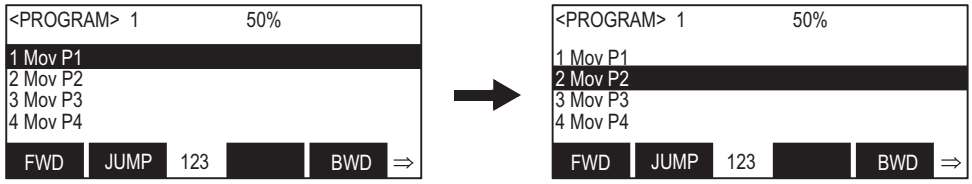
  – durch Loslassen oder Durchdrücken des Dreistufenschalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Betätigen Sie zur Überprüfung des Betriebs den Dreistufenschalter bis zur Mittelstellung und führen Sie das Programm im Schrittbetrieb aus.

  – durch Loslassen der Taste [F1](FWD)
    Die Ausführung des Schrittes wird unterbrochen. Die Servoversorgung wird nicht abgeschaltet. Betätigen Sie zum Fortsetzen der Programmausführung die Taste [F1](FWD).

Programm schrittweise ausführen (rückwärts)

Nach Abarbeitung einer Zeile wird die vorherige Zeile aufgerufen und ausgeführt. Die Funktion kann nur bei Interpolationsbefehlen verwendet werden. Die maximale Anzahl der Zeilen für die Programmausführung in Rückwärtsrichtung ist 4.

① Bei Betätigung der Funktionstaste, die der Funktion „BWD" zugeordnet ist, wird der vorhergehende Programmschritt ausgeführt.
Wird die Funktionstaste während der Bewegung losgelassen, stoppt der Roboter. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung und der Roboter stoppt ebenfalls.
Während der Ausführung eines Programmschritts leuchtet die LED der [START]-Taste am Steuergerät. Nach der Ausführung eines Schritts erlischt die LED und die LED der [STOP]-Taste leuchtet.
Der Cursor springt zum nächsten vorhergehenden Schritt, der einen Interpolationsbefehl enthält.
Bei jeder Betätigung der Taste [F4] wird der vorhergehende Schritt ausgeführt.



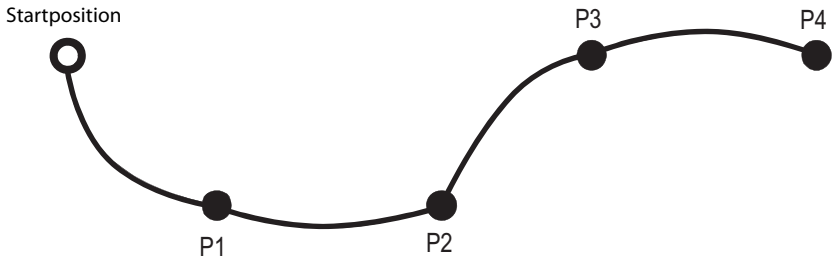
Betätigen Sie die Taste [F4], um das Programm schrittweise rückwärts auszuführen.

R001582E

HINWEIS

Wird in folgendem Programm der Schritt 4 im Schrittbetrieb vorwärts, danach die Position 1 im Schrittbetrieb rückwärts angefahren und dann ein weiterer Schritt rückwärts ausgeführt, kehrt der Roboter zur Startposition zurück. (Die Startposition ist die Position von der aus der Schritt 1 ausgeführt wurde.)

Programm
1 Mov P1
2 Mov P2
3 Mov P3
4 Mov P4



ACHTUNG:

Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.

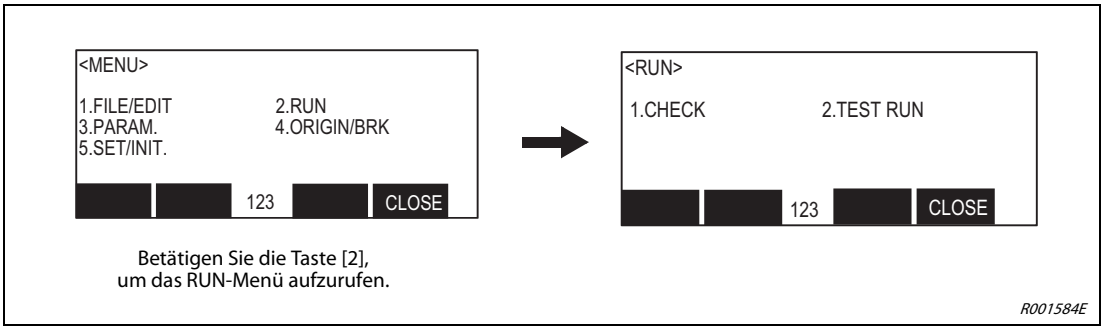
Beschreibung

- Stoppen des Roboters im Betrieb
  - durch Betätigung des NOT-HALT-Schalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Setzen Sie zur Überprüfung des Betriebs den Fehler zurück, schalten Sie die Servoversorgung wieder ein und führen Sie das Programm im Schrittbetrieb aus.
  - durch Loslassen oder Durchdrücken des Dreistufenschalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Betätigen Sie zur Überprüfung des Betriebs den Dreistufenschalter bis zur Mittelstellung und führen Sie das Programm im Schrittbetrieb aus.
  - durch Loslassen der Taste [F4](BWD)
    Die Ausführung des Schrittes wird unterbrochen. Die Servoversorgung wird nicht abgeschaltet. Betätigen Sie zum Fortsetzen der Programmausführung die Taste [F4](BWD).

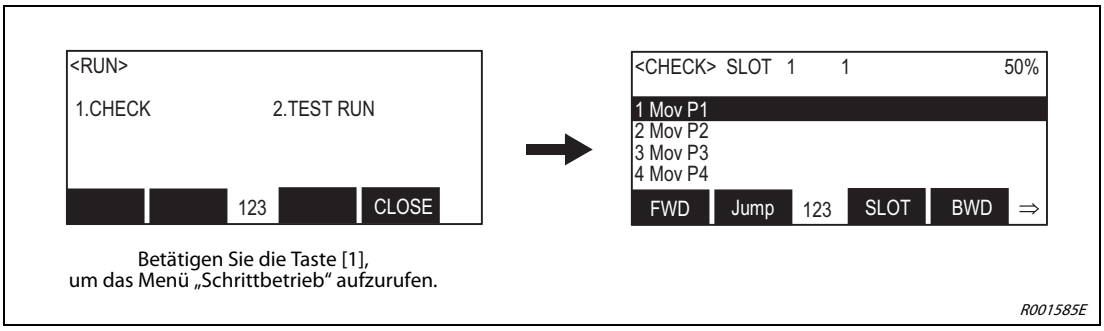
Programm eines anderen Programmplatzes schrittweise ausführen

Die schrittweise Ausführung eines Multitasking-Programms kann im Menü „Schrittbetrieb“ und muss nicht im Menü zur Programmeditierung ausgeführt werden.

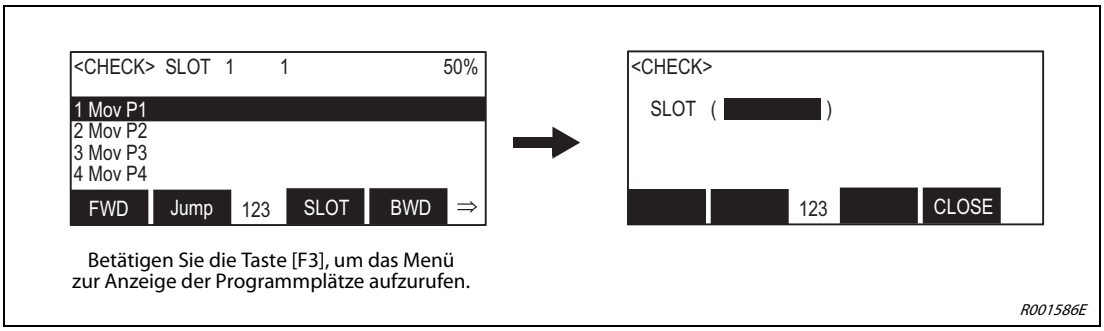
① Auswahl des RUN-Menüs
Betätigen Sie die Taste [2], um das RUN-Menü aufzurufen.



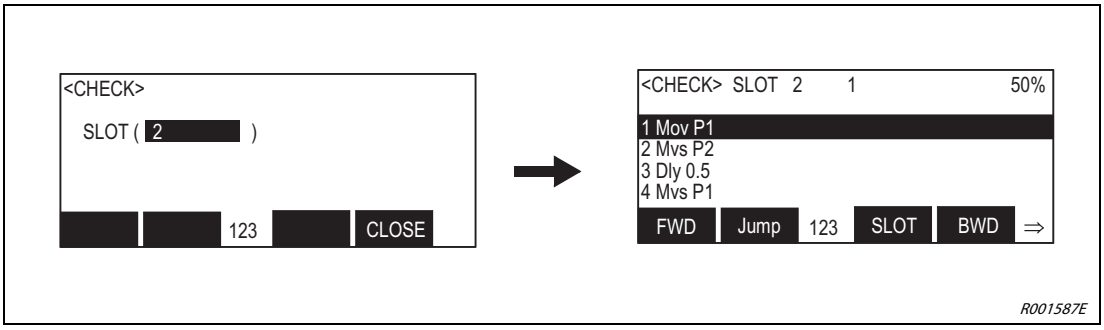
② Auswahl des Menüs „Schrittbetrieb“
Betätigen Sie die Taste [1], um das Menü „Schrittbetrieb“ aufzurufen.



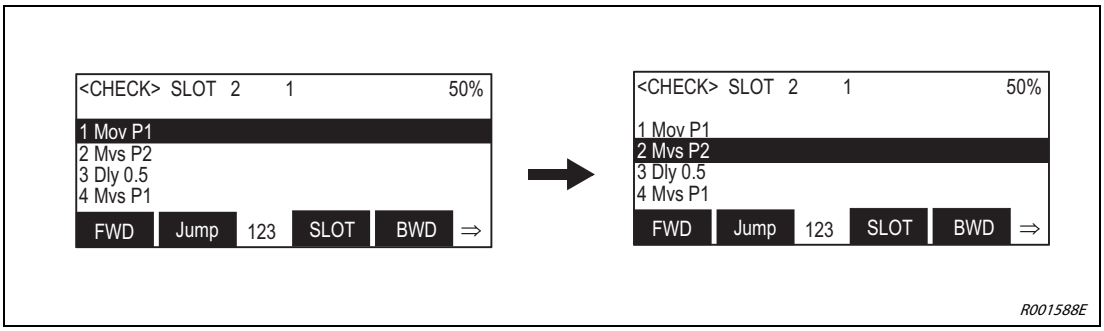
③ Auswahl des Programmplatzes
Betätigen Sie die Funktionstaste, die der Funktion „SLOT“ zugeordnet ist, um das Menü zur Anzeige der Programmplätze aufzurufen.



④ Geben Sie die Nummer des Programmplatzes ein und betätigen Sie die [EXE]-Taste. Das Programm des Programmplatzes wird angezeigt. (Im folgenden Beispiel ist das der Programmplatz 2)



⑤ Der Schrittbetrieb vorwärts und der Schrittbetrieb rückwärts können wie im Menü zur Programmeditierung ausgeführt werden. Führen Sie den Schrittbetrieb durch Betätigung der entsprechenden Funktionstaste aus.
Wird die Funktionstaste während der Bewegung losgelassen, stoppt der Roboter. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung und der Roboter stoppt ebenfalls.
Während der Ausführung eines Programmschritts leuchtet die LED der [START]-Taste am Steuergerät. Nach der Ausführung eines Schritts erlischt die LED und die LED der [STOP]-Taste leuchtet. Der Cursor springt in der gewählten Richtung zum nächsten Schritt, der einen Interpolationsbefehl enthält.



ACHTUNG:

Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.

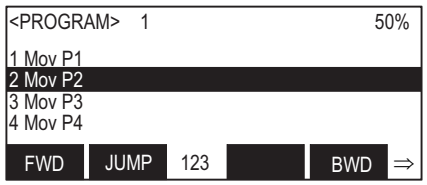
HINWEIS Mit Hilfe der Pfeiltasten oder der Funktion „Jump“ können Sie einen anderen Schritt aufrufen.

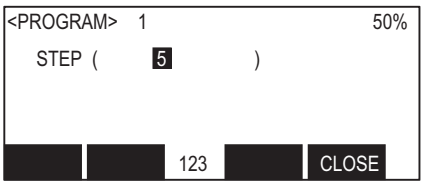
Sprung zu einer Programmzeile oder einem Programmschritt

Eine gewünschte Programmzeile oder ein Programmschritt kann aufgerufen werden.

Beispiel▽ In folgendem Beispiel erfolgt der Sprung zum Schritt 5.

① Auswahl des Schritts 5
Betätigen Sie die Funktionstaste, die der Funktion „JUMP“ zugeordnet ist. Betätigen Sie die Tasten [5] und [EXE]. Der Cursor springt zum Schritt 5. Der Schritt kann auch über die Pfeiltasten [↑] und [↓] aufgerufen werden.

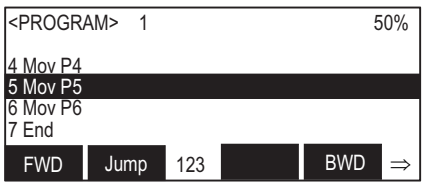


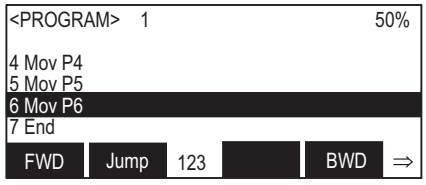


Betätigen Sie die Tasten [F2], [5] und [EXE], um zum Schritt 5 zu springen.

R001589E

② Ausführung des Schrittbetriebs
Betätigen Sie die Funktionstaste, die der Funktion „FWD“ zugeordnet ist, um den Schrittbetrieb ab Schritt 5 auszuführen.





Betätigen Sie die Taste [F1], um den Schrittbetrieb zu starten.

R001590E

△

ACHTUNG:

Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.

3.10 Automatikbetrieb

3.10.1 Geschwindigkeit einstellen

Die Geschwindigkeit kann über die Teaching Box oder das Steuergerät festgelegt werden. Die aktuelle Arbeitsgeschwindigkeit ergibt sich dabei aus:

Aktuelle Arbeitsgeschwindigkeit	=	Einstellwert über Steuergerät (Teaching Box)	×	Einstellwert im Programm

Einstellung über das Steuergerät (Modus Automatic)

① Betätigen Sie zweimal die Taste [CHNG DISP], um den Wert der Geschwindigkeitsübersteuerung „OVERRIDE“ anzuzeigen.

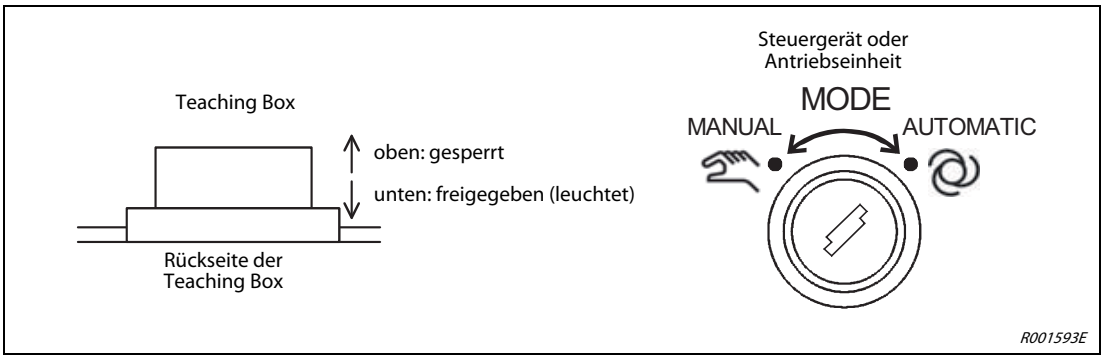
② Stellen Sie die Geschwindigkeitsübersteuerung über die Tasten [↑] und [↓] ein. Der Wert kann in folgenden Schritten eingestellt werden: 10 → 20 → 30 → 40 →50 → 60 → 70 → 80 → 90 → 100 %.

Einstellung über die Teaching Box (Modus Manual)

Eine Erhöhung der Geschwindigkeit erfolgt mit der Teaching Box über die Taste [OVRD ↑]. Der Wert kann in folgenden Schritten eingestellt werden: LOW → HIGH → 3 → 5 → 10 → 30 → 50 → 70 → 100 %. Eine Verminderung der Geschwindigkeit erfolgt mit der Teaching Box in der gleichen Schrittfolge über die Taste [OVRD ↓].

3.10.2 Auswahl der Programmnummer

① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „gesperrt“ und danach den [MODE]-Schalter des Steuergerätes auf „AUTOMATIC“.



② Betätigen Sie die [CHNG DISP]-Taste, um die Programmnummer anzuzeigen.

③ Wählen Sie die Programmnummer über die Tasten [↑] und [↓].

HINWEISE

Besteht ein Programmname aus mehr als 5 Zeichen, kann er nicht angezeigt werden. Bei einem Programmaufruf über ein externes Gerät erscheint „P----“ auf der Anzeige.

Die Auswahl eines Programmes über die Teaching Box ist ab Software-Version 1.3 der Teaching Box möglich (siehe auch Abschn. 3.14.6).

Wird ein Programm während der Ausführung gestoppt, muss zunächst die [Reset]-Taste betätigt werden, bevor ein neues Programm ausgewählt werden kann (siehe auch Abschn. 3.10.6).

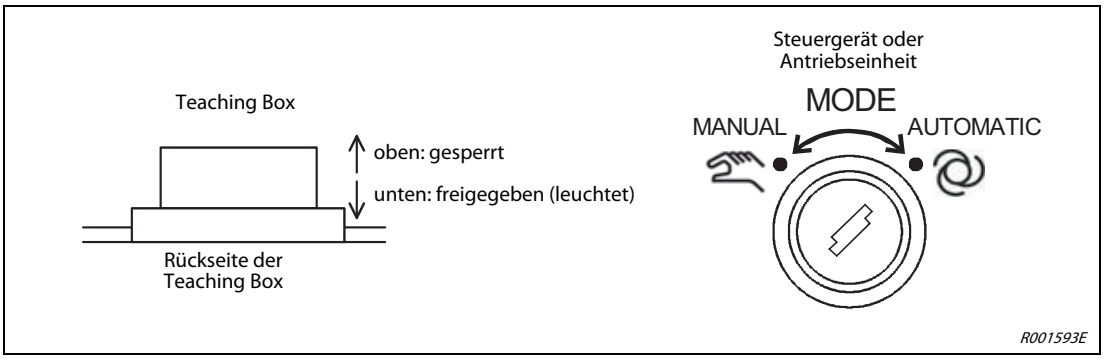
3.10.3 Starten des Automatikbetriebs

GEFAHR:

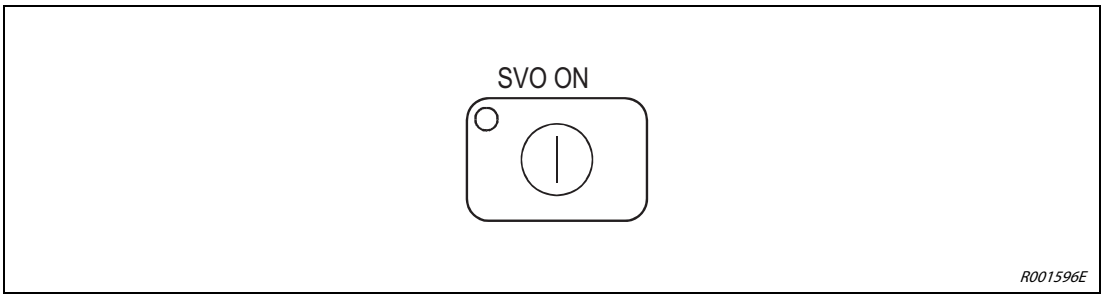
Stellen Sie vor einem Start des Automatikbetriebs sicher, dass die folgenden Bedingungen erfüllt sind:

- ***Es dürfen sich keine Personen im Umkreis des Roboters befinden.***
- ***Die Tür der Sicherheitsumzäunung muss geschlossen sein. Ein Öffnen der Tür muss zur sofortigen Unterbrechung des Roboterbetriebs führen.***
- ***Es dürfen sich keine Gegenstände innerhalb des Bewegungsbereiches des Roboters befinden, die zum Betrieb des Roboters nicht unbedingt notwendig sind (z. B.Werkzeuge usw.).***
- ***Überprüfen Sie die einwandfreie Funktion des Programms vor Ausführung des Automatikbetriebs im Schrittbetrieb.***

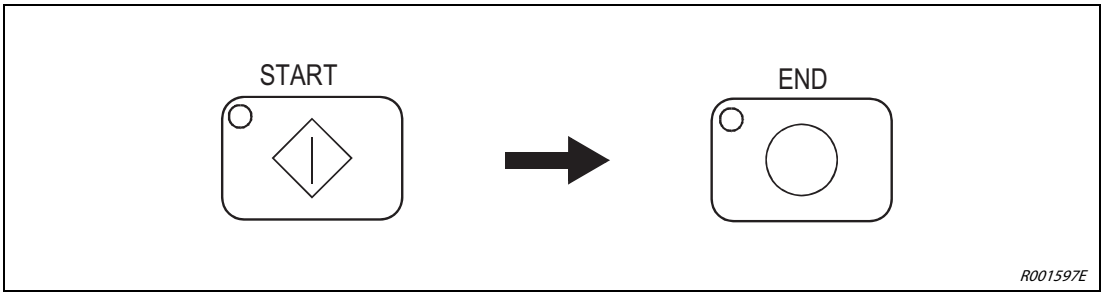
① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „gesperrt" und danach den [MODE]-Schalter des Steuergerätes auf „AUTOMATIC".



② Betätigen Sie die [SVO ON]-Taste, um die Servospannung einzuschalten. Die LED auf der [SVO ON]-Taste leuchtet.



③ Betätigen Sie die [START]-Taste, nachdem die LED auf der [SVO ON]-Taste leuchtet, um den kontinuierlichen Automatikbetrieb zu starten. Bei Betätigung der [END]-Taste stoppt das Programm nach Ausführung eines Zyklus. Die LED blinkt während der Ausführung des Zyklus.



GEFAHR:

- ***Prüfen Sie vor dem Starten des Automatikbetriebs, dass die korrekte Programmnummer ausgewählt wurde.***
- ***Unterbrechen Sie den Automatikbetrieb sofort durch Betätigung des NOT-HALT-Schalters, falls Ihnen Unregelmäßigkeiten auffallen.***
- ***Verfahren Sie den Roboter beim Start des Automatikbetriebs erst mit niedriger Geschwindigkeit. Steigern Sie die Geschwindigkeit allmählich und achten Sie darauf, dass keine Kollisionen mit umliegenden Einheiten auftreten.***

3.10.4 Stoppen des Automatikbetriebs

Der Automatikbetrieb kann über das Steuergerät oder über die Teaching Box durch Betätigung der [STOP]-Taste gestoppt werden.

Bei Betätigung der [STOP]-Taste wird das Programm unterbrochen und der Roboter bis zum Stillstand abgebremst.

HINWEIS Zur Ausführung der Stoppfunktion ist kein Betriebsrecht erforderlich.

3.10.5 Fortsetzung des Automatikbetriebs aus dem Stoppzustand

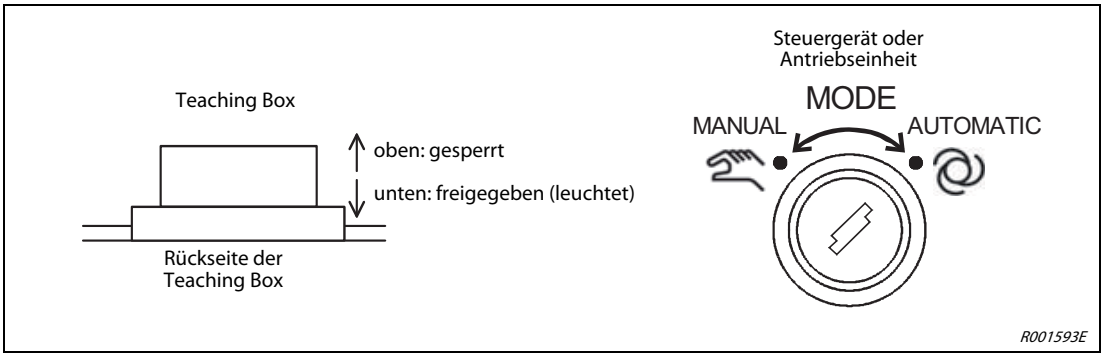
Zum Fortsetzen des Automatikbetriebs aus dem Stoppzustand führen Sie die Schritte aus, die im Absatz „Starten des Automatikbetriebs“ (siehe Seite 3-46) beschrieben werden.

3.10.6 Programm zurücksetzen

Der Stoppzustand des Programmes wird aufgehoben und die Programmsteuerung springt zum Programmanfang.

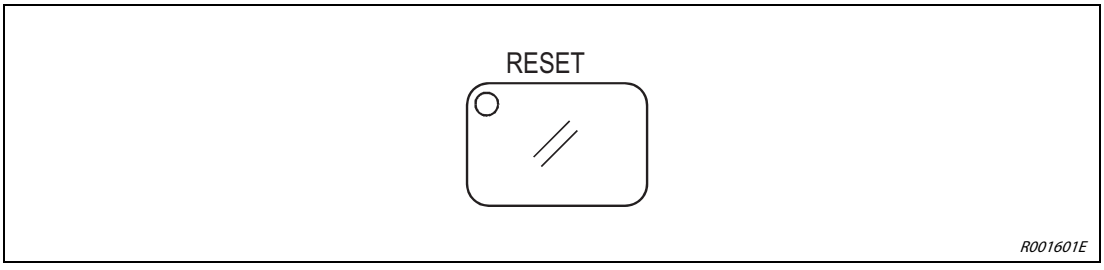
Rücksetzen über das Steuergerät

① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „gesperrt“ und danach den [MODE]-Schalter des Steuergerätes auf „AUTOMATIC“.



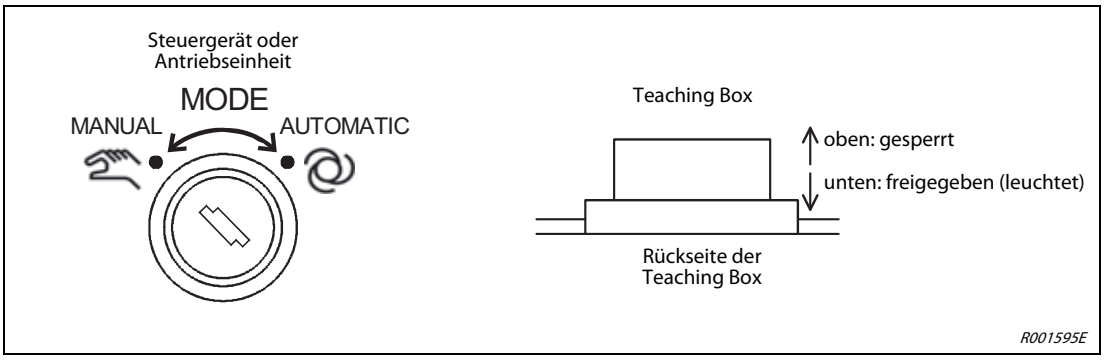
② Betätigen Sie die [CHNG DISP]-Taste, um die Programmnummer anzuzeigen.

③ Betätigen Sie die [RESET]-Taste. Die STOP-LED erlischt.



Rücksetzen über die Teaching Box

① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „MANUAL“ und danach den [ENABLE/ DISABLE]-Schalter der Teaching Box auf „freigegeben“.



② Betätigen Sie die [EXE]-Taste, während Sie die [RESET]-Taste gedrückt halten.

HINWEISE

Während der Programmausführung kann ein Programm nicht zurückgesetzt werden. Das Programm muss zuerst gestoppt werden. Vor Ausführung des Rücksetzvorgangs über das Steuergerät muss die entsprechende Programmnummer im Display angezeigt werden.

Die LED der [STOP]-Taste erlischt, wenn das Programm zurückgesetzt ist.

3.11 Servospannung ein-/ausschalten

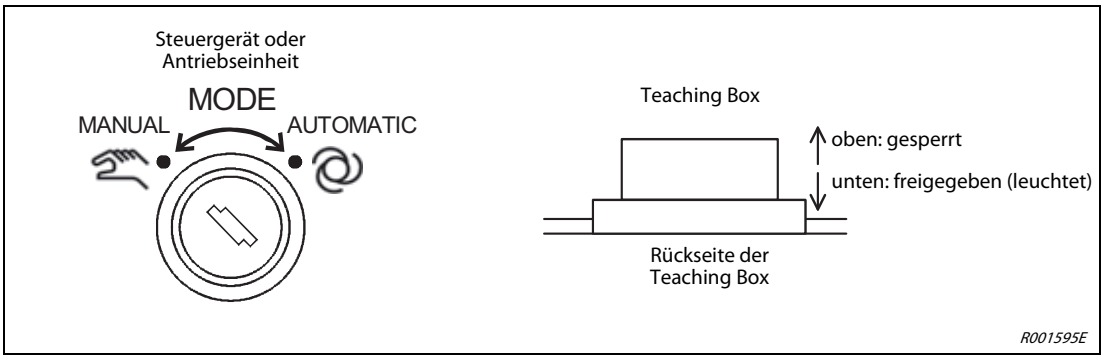
Aus Sicherheitsgründen kann die Servospannung im Teach-Modus nur bei betätigtem Dreistufenschalter eingeschaltet werden. Betätigen Sie den Dreistufenschalter bis zur Mittelstellung, bevor Sie die Servospannung einschalten.

HINWEIS

Die Bremsen werden automatisch aktiviert, wenn die Servospannung ausgeschaltet wird. Je nach Robotertyp verfügen nicht alle Achsen über Bremsen.

Servospannung über die [Servo]-Taste der Teaching Box einschalten

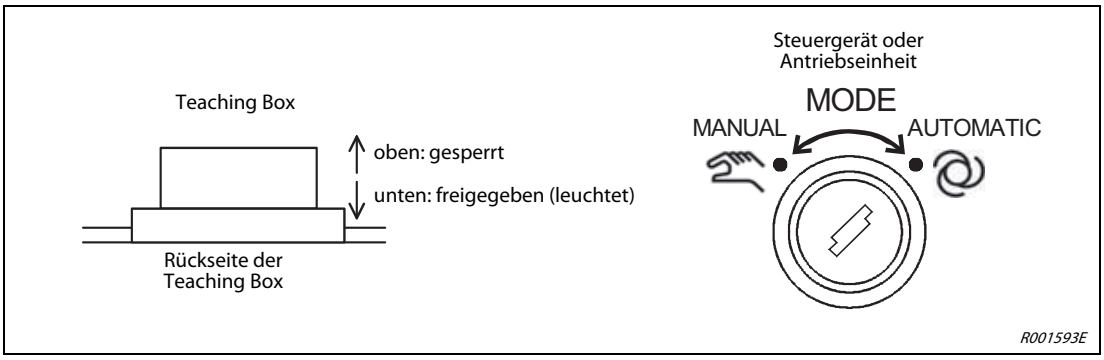
① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „MANUAL“ und danach den [ENABLE/DISABLE]-Schalter der Teaching Box auf „freigegeben“.



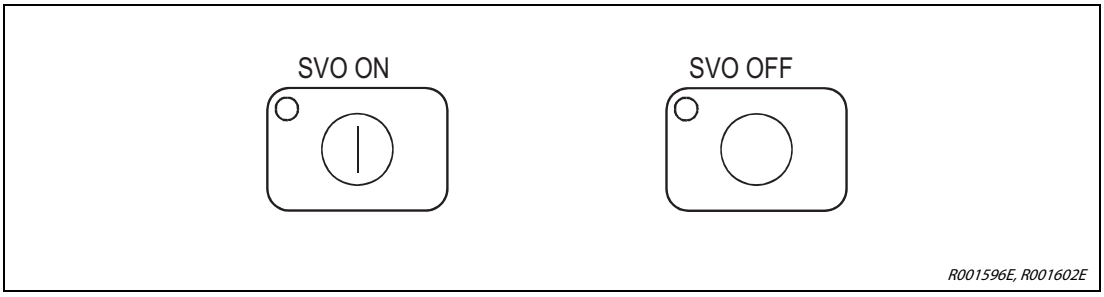
② Betätigen Sie die [SERVO]-Taste, um die Servospannung einzuschalten. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung.

Servospannung über das Steuergerät einschalten

① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „gesperrt“ und danach den [MODE]-Schalter des Steuergerätes auf „AUTOMATIC“.



② Betätigen Sie die [SVO ON]-Taste, um die Servospannung einzuschalten. Die LED auf der [SVO ON]-Taste leuchtet. Betätigen Sie die [SVO OFF]-Taste, um die Servospannung auszuschalten. Die LED auf der [SVO OFF]-Taste leuchtet.



3.12 Fehler zurücksetzen

Fehler über das Steuergerät zurücksetzen

① Betätigen Sie die [RESET]-Taste, um den Fehler zurückzusetzen.

Fehler über die Teaching Box zurücksetzen

① Betätigen Sie die [RESET]-Taste, um den Fehler zurückzusetzen.

HINWEIS

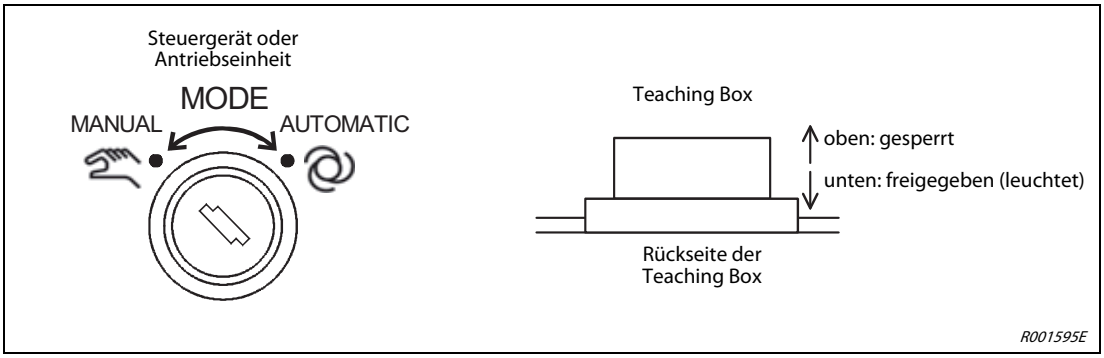
Fehler, wie z.B. H0070, lassen sich unabhängig vom Betriebsmodus des Steuergeräts bzw. der Teaching Box jederzeit zurücksetzen.

3.13 Fehler temporär zurücksetzen

In Abhängigkeit des Robotertyps können Fehler auftreten, die sich nicht zurücksetzen lassen. Das kann z.B. dann der Fall sein, wenn die Koordinaten einer Achse außerhalb des Bewegungsbereiches des Roboters liegen. Die Servospannung kann dann nicht mehr eingeschaltet werden, und es ist nicht möglich, den Roboter im JOG-Betrieb in den zulässigen Bereich zu bewegen. Hier muss der Fehler temporär zurückgesetzt werden. Anschließend kann der Roboter im JOG-Betrieb wieder in den zulässigen Bewegungsbereich verfahren werden.

Führen Sie folgende Schritte aus, um einen Fehler temporär zurückzusetzen:

① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „MANUAL“ und danach den [ENABLE/ DISABLE]-Schalter der Teaching Box auf „freigegeben“.



② Halten Sie den Dreistufenschalter in Mittelstellung und betätigen Sie anschließend die [RESET]-Taste, während Sie die [SERVO]-Taste gedrückt halten.

HINWEIS

Die oben genannten Schritte dienen zum temporären Zurücksetzen eines Fehlers. Die [RESET]-Taste muss auch bei der Ausführung des JOG-Betriebes weiterhin gehalten werden. Wird die Taste losgelassen, tritt der Fehler erneut auf.

3.14 Programmverwaltungsfunktionen

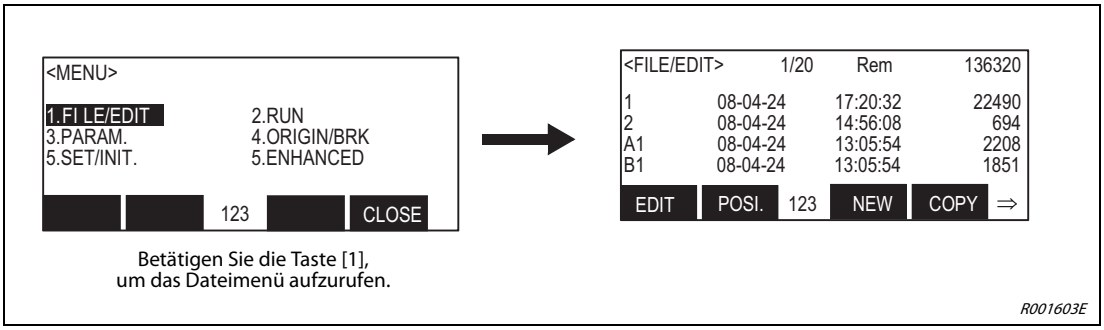
Im Folgenden werden die aufgelisteten Programmverwaltungsfunktionen beschrieben:

- Programmverzeichnis anzeigen
- Programm kopieren
- Programmnamen ändern
- Programm löschen
- Programm schützen
- Programm auswählen

3.14.1 Programmverzeichnis anzeigen

Die Funktion ermöglicht die Anzeige der Programme, die im Steuergerät gespeichert sind, und von deren Eigenschaften.

① Aufruf des Dateimenüs
Betätigen Sie die Taste [1] im Hauptmenü, um das Dateimenü aufzurufen. Das Programmverzeichnis wird angezeigt.

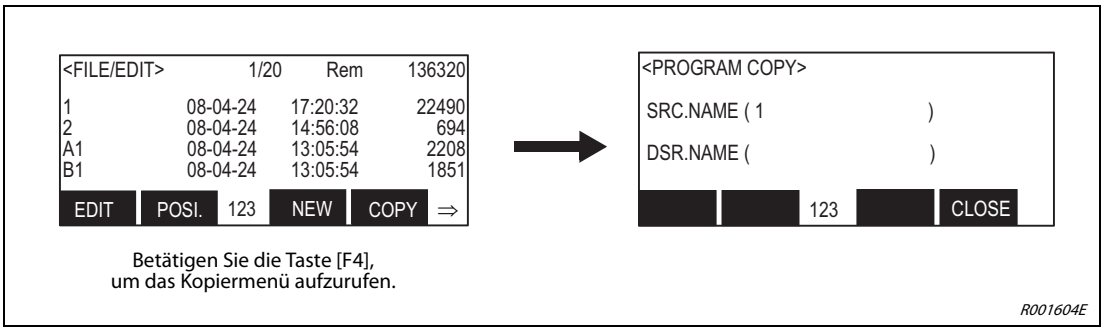


Das Dateimenü kann auch aufgerufen werden, indem Sie den Cursor mit den Pfeiltasten [↑] und [↓] zum Punkt „1. FILE/EDIT“ bewegen und die [EXE]-Taste betätigen. So können Sie auch ein Programm auswählen.

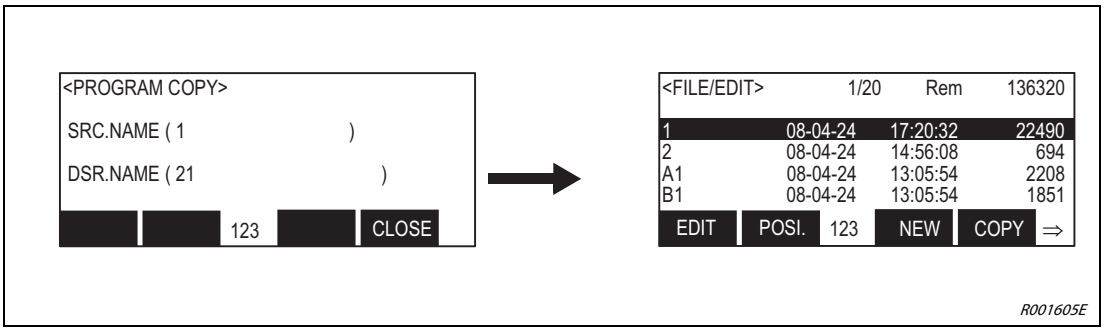
Wählen Sie die Funktionen „EDIT“, POSI.“, „NEW“ oder „COPY“ über die entsprechende Funktionstaste aus. Betätigen Sie die Taste [FUNCTION], um die Funktionen „RENAME“, „DELETE“, „PRTCT“ und „CLOSE“ aufzurufen.

3.14.2 Programm kopieren

① Aufruf des Kopiermenüs
Betätigen Sie die Funktionstaste, die der Funktion „COPY“ zugeordnet ist, um das Kopiermenü aufzurufen.



② Festlegung des zu kopierenden Programms und Kopiervorgang
Die zuvor im Dateimenü ausgewählte Quelldatei wird in Klammern hinter „SRC. NAME“ angezeigt (hier Programm 1). Wählen Sie das Programm über die Pfeiltasten aus.
Geben Sie den Namen des Zielprogramms in die Klammern hinter „DSR. NAME“ ein und betätigen Sie die [EXE]-Taste.



HINWEISE

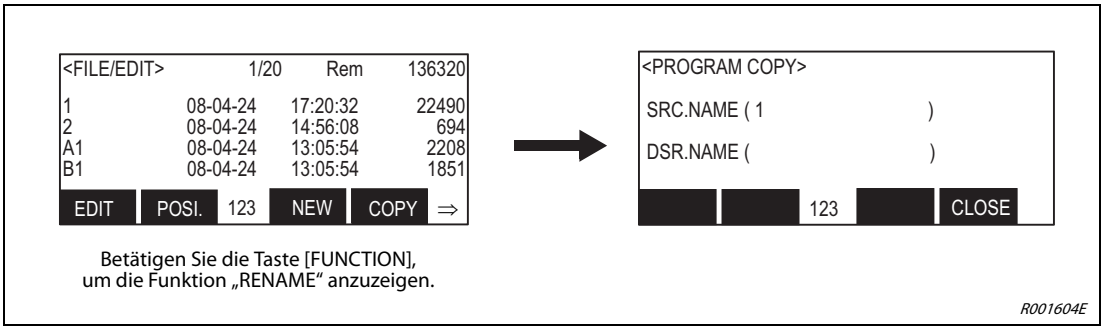
Es wird eine Fehlermeldung angezeigt, wenn derselbe Programmname zweimal angegeben wird, d.h. wenn Quelle und Ziel eines Kopiervorgangs identisch sind.

Der Schutzstatus von Programmen und Variablen wird nicht mitkopiert.

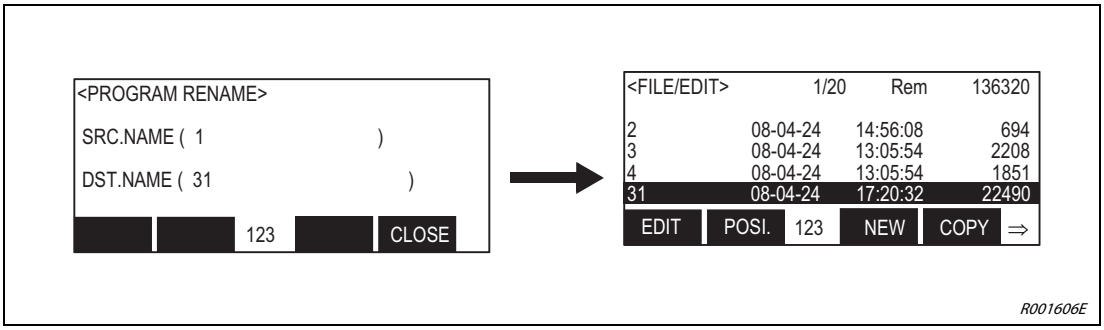
Es wird eine Fehlermeldung angezeigt, wenn das Zielprogramm belegt ist.

3.14.3 Programmnamen ändern

① Aufruf des Menüs zum Ändern des Programmnamens
Betätigen Sie die Funktionstaste, die der Funktion „RENAME“ zugeordnet ist, um das Menü zum Ändern des Programmnamens aufzurufen. Wird die Funktion „RENAME“ nicht angezeigt, betätigen Sie die Taste [FUNCTION].



② Festlegung des Programms, dessen Name geändert werden soll
Die zuvor im Dateimenü ausgewählte Datei wird in Klammern hinter „SCR. NAME“ angezeigt (hier Programm 1). Wählen Sie das Programm über die Pfeiltasten aus.
Geben Sie den neuen Namen in die Klammern hinter „DST. NAME“ ein und betätigen Sie die [EXE]-Taste.



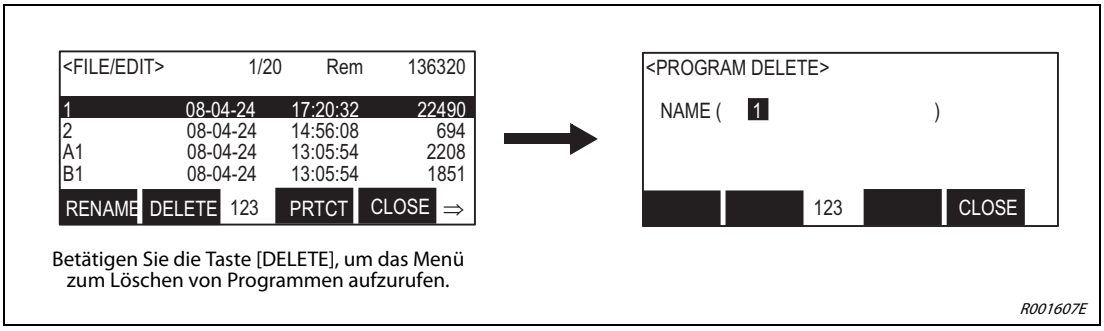
HINWEISE

Es wird eine Fehlermeldung angezeigt, wenn derselbe Programmname zweimal angegeben wird.

Der Programmname eines Programms mit geschützten Befehlen kann nicht geändert werden. Schalten Sie gegebenfalls die Schutzfunktion aus.

3.14.4 Programm löschen

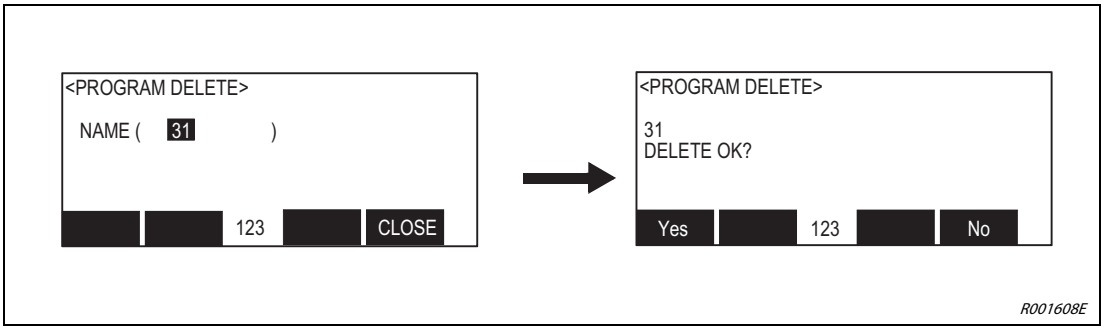
① Aufruf des Menüs zum Löschen von Programmen
Betätigen Sie die Funktionstaste, die der Funktion „DELETE“ zugeordnet ist, um das Menü zum Löschen von Programm aufzurufen. Wird die Funktion „DELETE“ nicht angezeigt, betätigen Sie die Taste [FUNCTION].



Betätigen Sie die Taste [DELETE], um das Menü zum Löschen von Programmen aufzurufen.

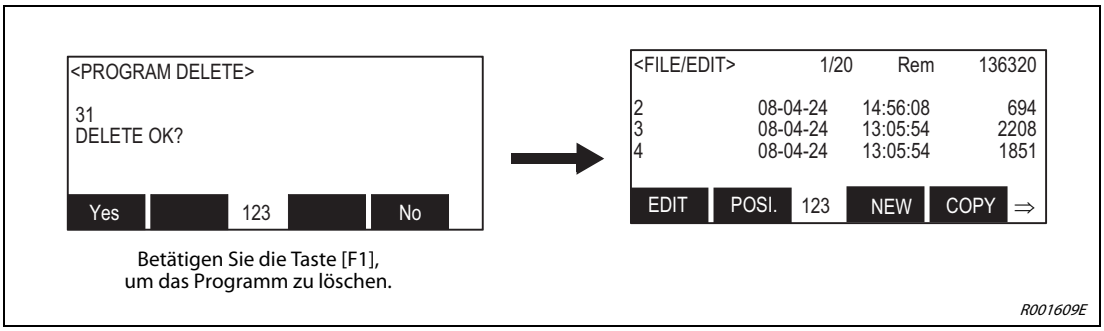
R001607E

② Festlegung des Programms, das gelöscht werden soll
Die zuvor im Dateimenü ausgewählte Datei wird in Klammern hinter „NAME“ angezeigt. Wählen Sie das Programm, das Sie löschen möchten, über die Pfeiltasten aus oder geben Sie den Namen über die Tastatur ein.
Betätigen Sie die [EXE]-Taste. Das Bestätigungsmenü erscheint.



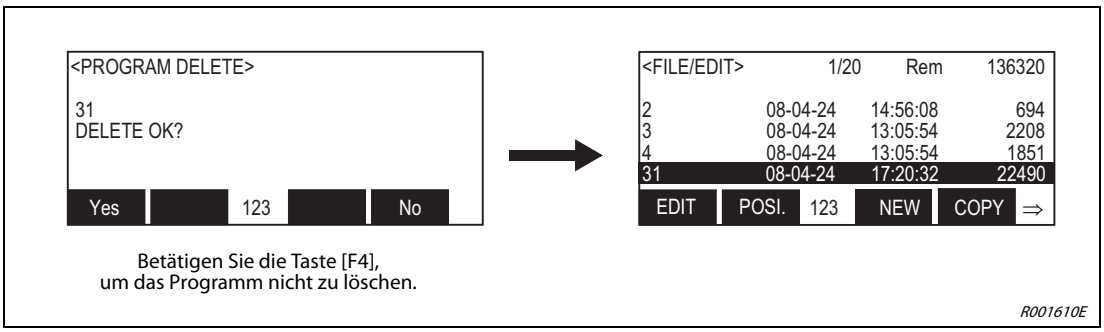
R001608E

③ Löschen des Programms
Betätigen Sie die Funktionstaste, die der Funktion „Yes“ zugeordnet ist, um das Programm zu löschen und zum Programmverzeichnis zurückzukehren.
Möchten Sie das Programm nicht löschen, betätigen Sie die Taste, die der Funktion „No“ zugeordnet ist. Das Menü zum Löschen des Programms erscheint.



Betätigen Sie die Taste [F1],
um das Programm zu löschen.

R001609E



Betätigen Sie die Taste [F4],
um das Programm nicht zu löschen.

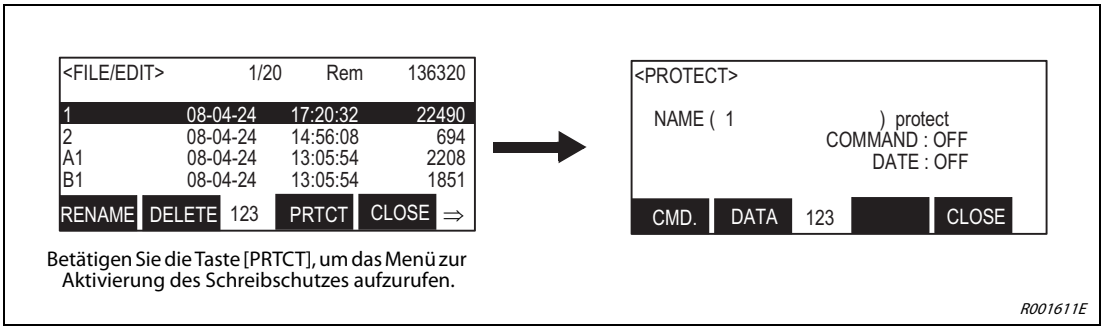
R001610E

HINWEIS

Ein Programm mit geschützten Befehlen kann nicht gelöscht werden. Schalten Sie gegebenfalls die Schutzfunktion aus.

3.14.5 Programm schützen

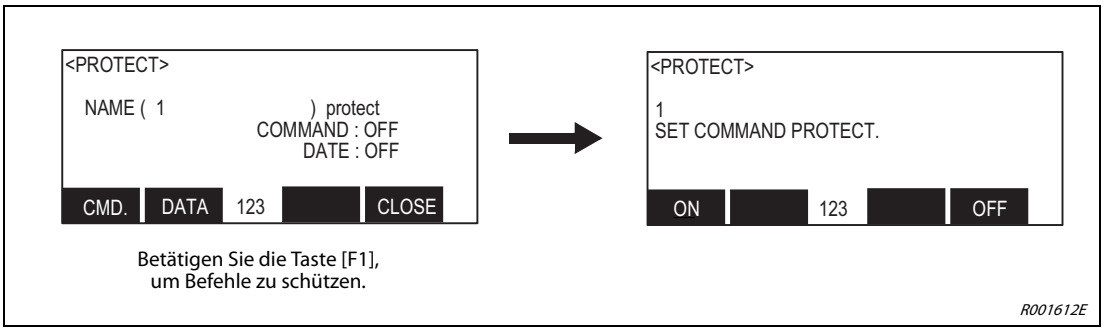
① Aufruf des Menüs zur Aktivierung des Schreibschutzes
Betätigen Sie die Funktionstaste, die der Funktion „PRTCT“ zugeordnet ist, um das Menü zur Aktivierung des Schreibschutzes aufzurufen. Wird die Funktion „PRTCT“ nicht angezeigt, betätigen Sie die Taste [FUNCTION].



Betätigen Sie die Taste [PRTCT], um das Menü zur Aktivierung des Schreibschutzes aufzurufen.

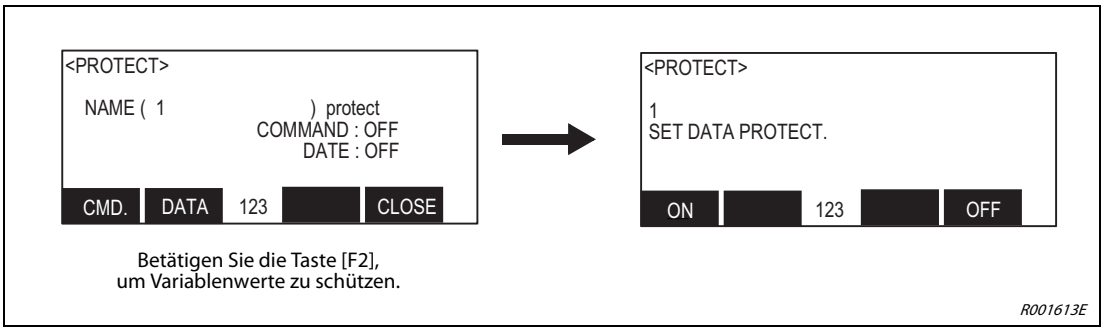
R001611E

② Festlegung des Schreibschutzes
Befehle und Daten (Variablenwerte) können separat geschützt werden. Wählen Sie die Funktion „CMD.“ oder „DATA“ über die entsprechende Funktionstaste.



Betätigen Sie die Taste [F1], um Befehle zu schützen.

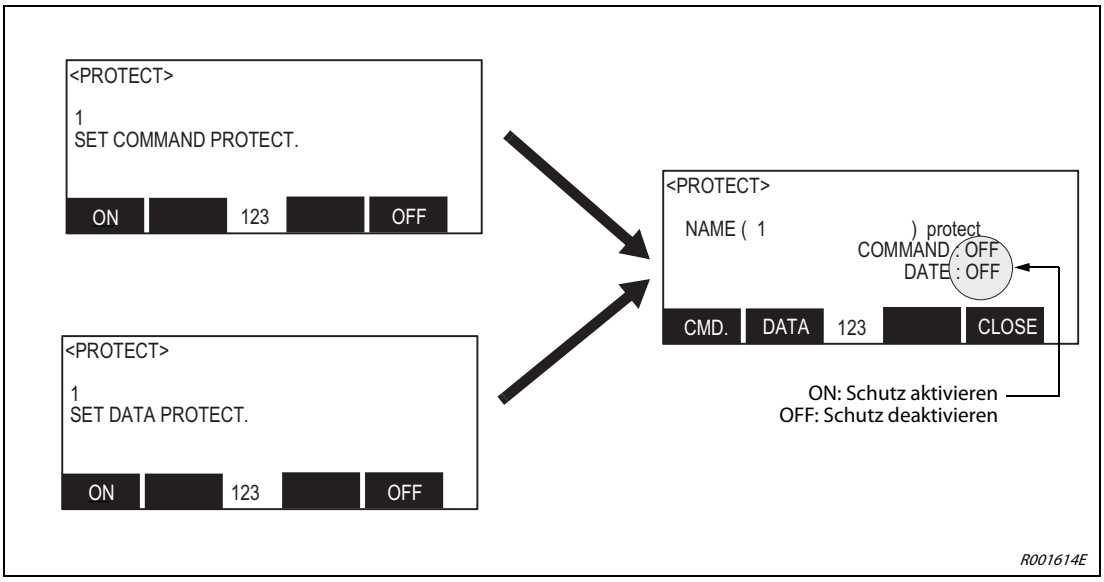
R001612E



Betätigen Sie die Taste [F2], um Variablenwerte zu schützen.

R001613E

③ Aktivieren sie den Schreibschutz über die Funktion „ON", deaktivieren Sie den Schreibschutz über die Funktion „OFF".



Schutz von Befehlen

- Programme werden gegen die Operationen DELETE, RENAME und Programmänderungen geschützt.
- Beim Kopieren eines Programms wird der Schutzstatus nicht mitkopiert.
- Beim Löschen des Speichers über „INITIALIZATION" (siehe Abschn. 3.20.1) wird der Schutzstatus ignoriert und das Programm gelöscht.

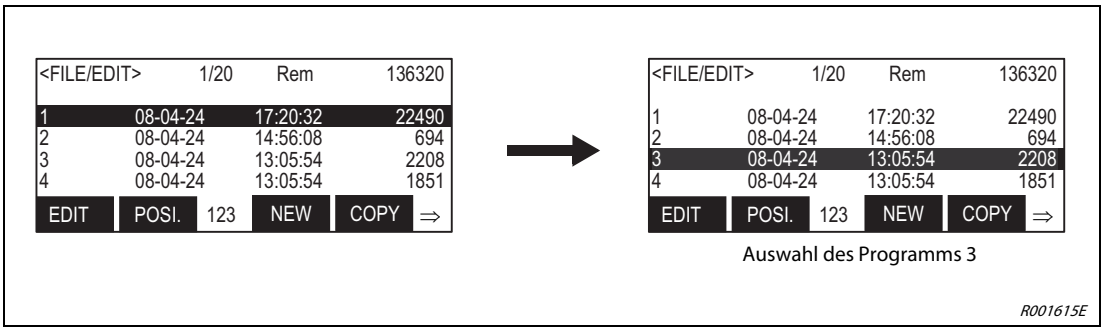
Schutz von Daten

- Variablen werden gegen das Überschreiben durch Positionsdaten, gegen Änderungen und gegen Substitutionen bei fehlerhafter Programmausführung geschützt.
- Beim Kopieren eines Programms wird der Schutzstatus nicht mitkopiert.
- Beim Löschen des Speichers über „INITIALIZATION" (siehe Abschn. 3.20.1) wird der Schutzstatus ignoriert und die Variablen gelöscht.

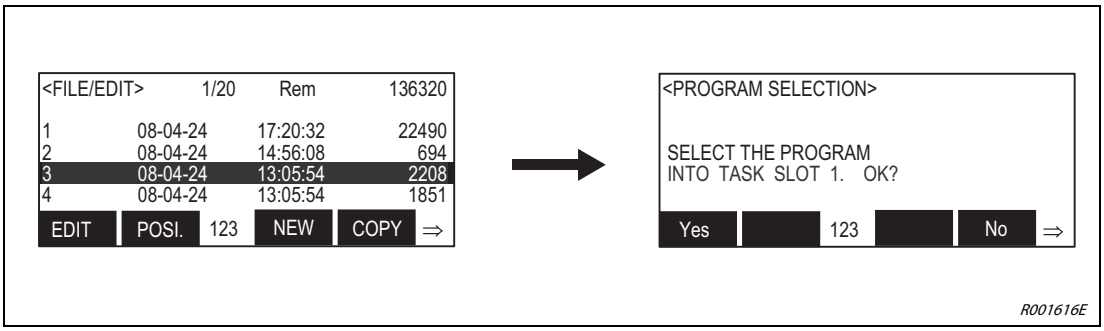
3.14.6 Programm auswählen

Diese Funktion steht ab Software-Version 1.3 der Teaching Box zur Verfügung.
Das Programm kann für den Schritt- oder den Automatikbetrieb ausgewählt werden.
Die Funktion entspricht der im Abschn. 3.10.2 „Auswahl der Programmnummer" beschriebenen Auswahl über das Steuergerät.
Nach der Auswahl des Programms erscheint die Programmnummer auf der Anzeige „STATUS NUMBER" des Steuergeräts.
Gehen Sie zur Auswahl eines Programmes wie folgt vor.

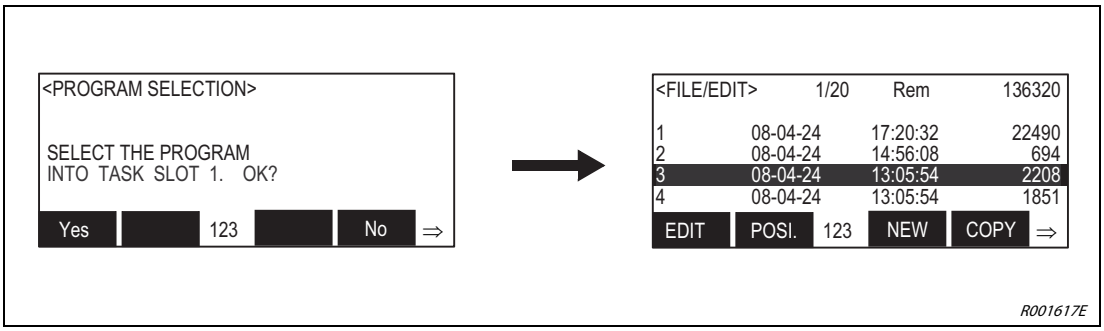
① Auswahl des Programms
Bewegen Sie den Cursor über die Pfeiltasten zum gewünschten Programm.



② Auswahl der Programmnummer
Betätigen Sie die Tasten [FUNCTION] und [EXE]. Das Bestätigungsmenü erscheint.



③ Überprüfen Sie die ausgewählte Programmnummer. Betätigen Sie die Funktionstaste, die der Funktion „Yes" zugeordnet ist, um das Programm auszuwählen. Betätigen Sie die Funktionstaste, die der Funktion „No" zugeordnet ist, um den Vorgang abzubrechen und zur Anzeige des Programmverzeichnisses zurückzukehren.



④ Die Auswahl des Programms ist abgeschlossen.

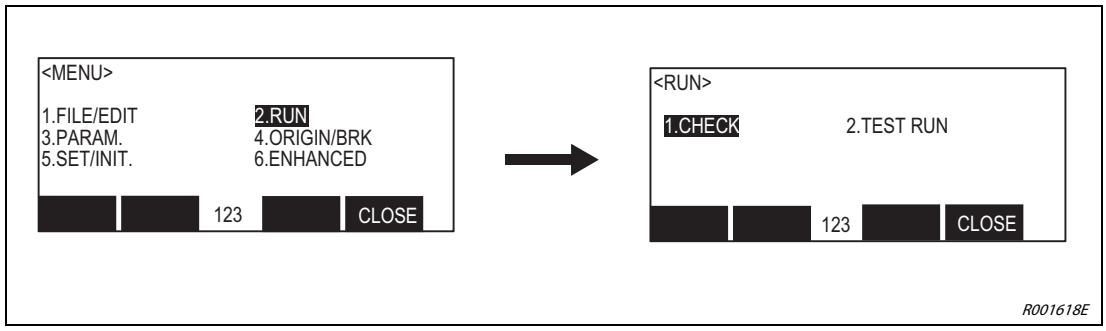
3.15 RUN-Menü

- Anzeige der Zeile, die gerade ausgeführt wird
  - Anzeige der Programmzeile, die gerade ausgeführt wird
  - Programm schrittweise ausführen
- Testbetrieb
  - Anzeige des Namens des gewählten Programms und des Schritts, der gerade ausgeführt wird
  - Wechsel vom kontinuierlichen in den Zyklus-Stopp-Modus

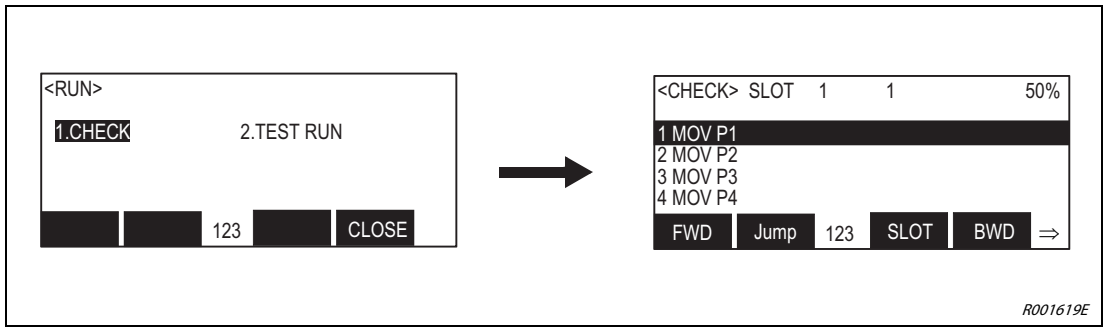
3.15.1 Anzeige der Zeile, die gerade ausgeführt wird

Auswahl des Betriebsmenüs

① Betätigen Sie die Taste [2] im Hauptmenü, um das RUN-Menü aufzurufen.

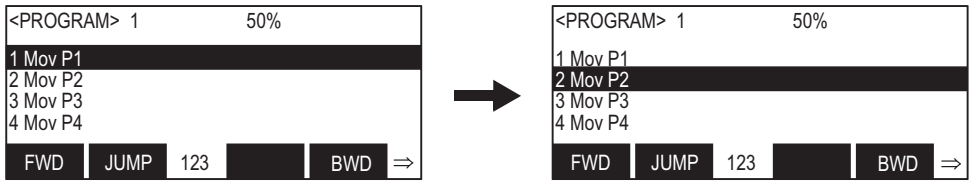


② Betätigen Sie die Taste [1], um das Betriebsmenü aufzurufen. Das Programm im Programmplatz 1 wird angezeigt. Hinter der Anzeige des Programmnamens erscheint die Nummer des Programmplatzes. Während der Ausführung des Programms wird die aktuell ausgeführte Zeile markiert dargestellt.



Programm schrittweise ausführen (vorwärts)

① Betätigen Sie die Taste [F1](FWD), um die Roboterbewegung zu starten.
Wird die [F1]-Taste während der Bewegung losgelassen , stoppt der Roboter. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung und der Roboter stoppt ebenfalls.
Während der Ausführung eines Programmschritts leuchtet die LED der [START]-Taste am Steuergerät. Nach der Ausführung eines Schritts erlischt die LED und die LED der [STOP]-Taste leuchtet.
Wird die [F1]-Taste losgelassen, springt der Cursor zum nächsten Schritt.
Bei jeder Betätigung der [F1]-Taste wird der nächste Schritt ausgeführt.



Betätigen Sie die Taste [F1],
um den Schrittbetrieb zu starten.

R001581E

ACHTUNG:

Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.

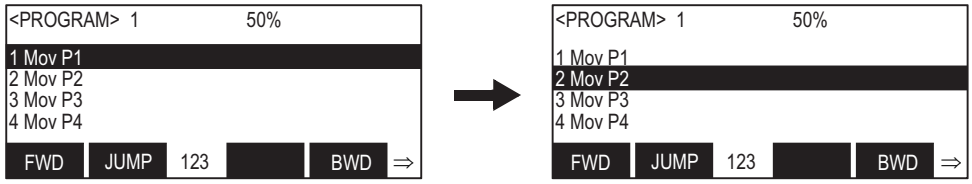
Beschreibung

- Schrittbetrieb
  Im Schrittbetrieb wird das Roboterprogramm zeilenweise abgearbeitet. Die Verfahrgeschwindigkeit ist niedrig und der Roboter stoppt nach Abarbeitung jeder Zeile, um eine Überprüfung der Programmfunktionen und der Verfahrbewegung zu ermöglichen. Während des Schrittbetriebs leuchtet die grüne LED des [START]-Tasters am Steuergerät.
  Bei Ausführung eines End- oder Hlt-Befehls wird der Schrittbetrieb abgebrochen.
- Aufruf eines anderen Schritts
  Mit Hilfe der Pfeiltasten oder der Funktion „Jump“ können Sie einen anderen Schritt aufrufen.
- Stoppen des Roboters im Betrieb
  - durch Betätigung des NOT-HALT-Schalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Setzen Sie zur Überprüfung des Betriebs den Fehler zurück, schalten Sie die Servoversorgung wieder ein und führen Sie das Programm im Schrittbetrieb aus.
  - durch Loslassen oder Durchdrücken des Dreistufenschalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Betätigen Sie zur Überprüfung des Betriebs den Dreistufenschalter bis zur Mittelstellung und führen Sie das Programm im Schrittbetrieb aus.
  - durch Loslassen der Taste [F1](FWD)
    Die Ausführung des Schrittes wird unterbrochen. Die Servoversorgung wird nicht abgeschaltet. Betätigen Sie zum Fortsetzen der Programmausführung die Taste [F1](FWD).

Programm schrittweise ausführen (rückwärts)

Nach Abarbeitung einer Zeile wird die vorherige Zeile aufgerufen und ausgeführt. Die Funktion kann nur bei Interpolationsbefehlen verwendet werden. Die maximale Anzahl der Zeilen für die Programmausführung in Rückwärtsrichtung ist 4.

① Bei Betätigung der Funktionstaste, die der Funktion „BWD" zugeordnet ist, wird der vorhergehende Programmschritt ausgeführt.
Wird die Funktionstaste während der Bewegung losgelassen, stoppt der Roboter. Wird der Dreistufenschalter ganz durchgedrückt oder losgelassen, erfolgt eine Abschaltung der Servospannung und der Roboter stoppt ebenfalls.
Während der Ausführung eines Programmschritts leuchtet die LED der [START]-Taste am Steuergerät. Nach der Ausführung eines Schritts erlischt die LED und die LED der [STOP]-Taste leuchtet.
Der Cursor springt zum nächsten vorhergehenden Schritt, der einen Interpolationsbefehl enthält.
Bei jeder Betätigung der Taste [F4] wird der vorhergehende Schritt ausgeführt.



Betätigen Sie die Taste [F4], um das Programm schrittweise rückwärts auszuführen.

R001582E

ACHTUNG:

Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.

Beschreibung

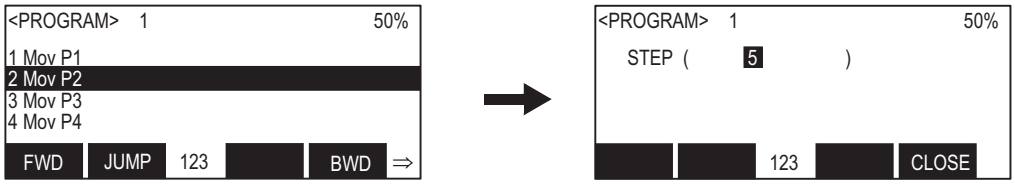
- Stoppen des Roboters im Betrieb
  - durch Betätigung des NOT-HALT-Schalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Setzen Sie zur Überprüfung des Betriebs den Fehler zurück, schalten Sie die Servoversorgung wieder ein und führen Sie das Programm im Schrittbetrieb aus.
  - durch Loslassen oder Durchdrücken des Dreistufenschalters
    Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Betätigen Sie zur Überprüfung des Betriebs den Dreistufenschalter bis zur Mittelstellung und führen Sie das Programm im Schrittbetrieb aus.
  - durch Loslassen der Taste [F4](BWD)
    Die Ausführung des Schrittes wird unterbrochen. Die Servoversorgung wird nicht abgeschaltet. Betätigen Sie zum Fortsetzen der Programmausführung die Taste [F4](BWD).

Sprung zu einer Programmzeile oder einem Programmschritt

Eine gewünschte Programmzeile oder ein Programmschritt kann aufgerufen werden.

Beispiel▽ In folgendem Beispiel erfolgt der Sprung zum Schritt 5.

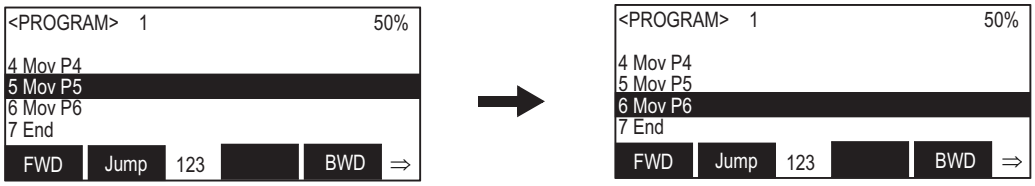
① Auswahl des Schritts 5
Betätigen Sie die Funktionstaste, die der Funktion „JUMP“ zugeordnet ist. Betätigen Sie die Tasten [5] und [EXE]. Der Cursor springt zum Schritt 5.



Betätigen Sie die Tasten [F2], [5] und [EXE], um zum Schritt 5 zu springen.

R001589E

② Ausführung des Schrittbetriebs
Betätigen Sie die Funktionstaste, die der Funktion „FWD“ zugeordnet ist, um den Schrittbetrieb ab Schritt 5 auszuführen.



Betätigen Sie die Taste [F1], um den Schrittbetrieb zu starten.

R001590E

△

ACHTUNG:

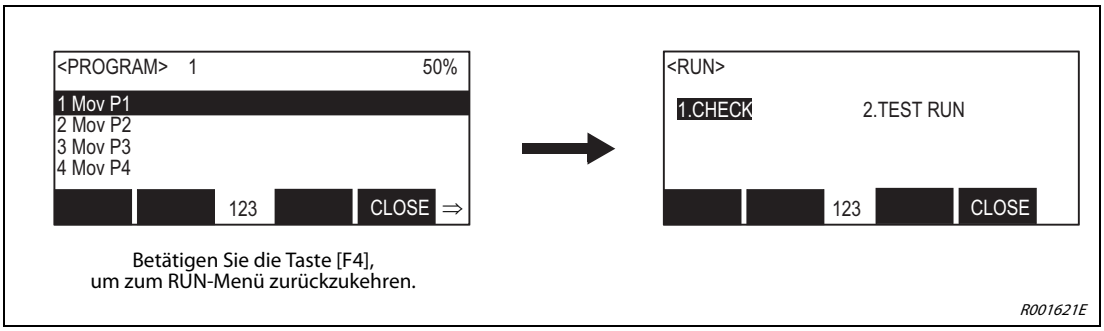
Achten Sie bei der Ausführung des Programms im Automatikbetrieb genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf, betätigen Sie den NOT-HALT-Schalter, um den Roboter zu stoppen.

Programm eines anderen Programmplatzes schrittweise ausführen

Die schrittweise Ausführung eines Multitasking-Programms kann im Menü „Schrittbetrieb“ und muss nicht im Menü zur Programmeditierung ausgeführt werden. Eine detaillierte Beschreibung der Funktion finden Sie auf Seite 3-41 „Programm eines anderen Programmplatzes schrittweise ausführen“.

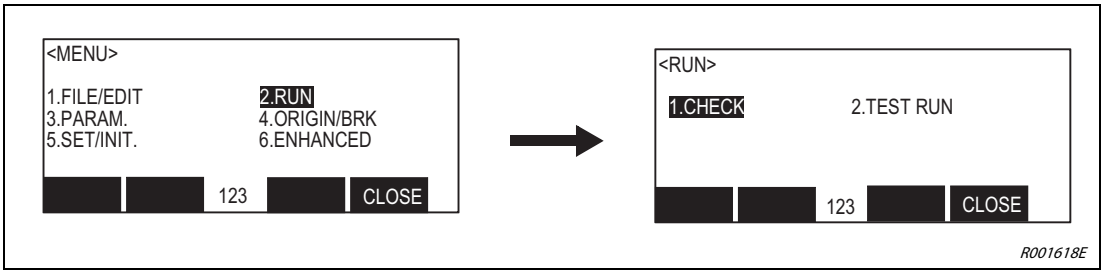
Beenden des Schrittbetriebs

① Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum RUN-Menü zurückzukehren.

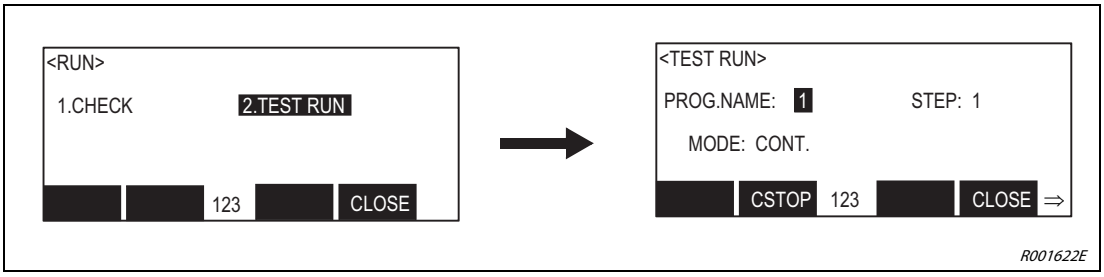


3.16 Testbetrieb

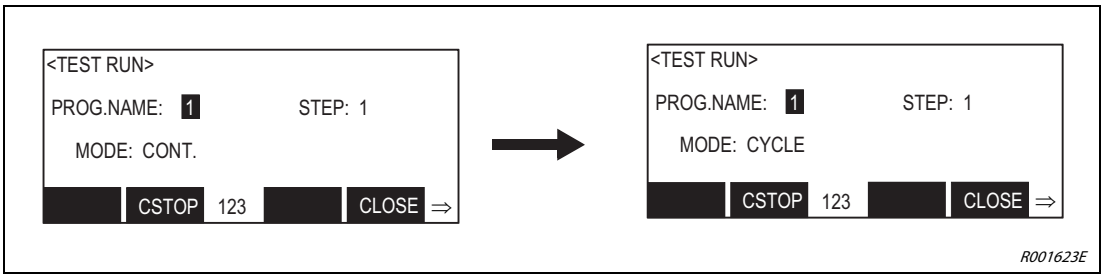
① Betätigen Sie die Taste [2] im Hauptmenü, um das RUN-Menü aufzurufen.



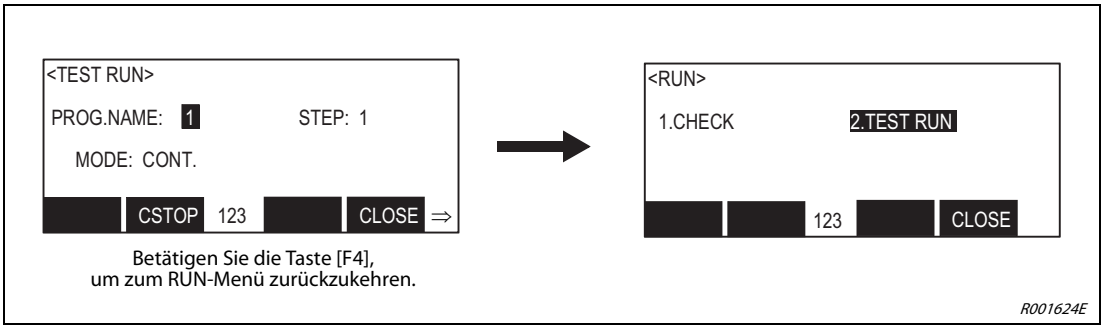
② Betätigen Sie die Taste [2] im RUN-Menü, um das Menü „Testbetrieb" aufzurufen. Der Programmname, die aktuelle Schrittnummer und der Betriebsmodus werden angezeigt.



③ Betätigen Sie während der Ausführung des Programms die Funktionstaste, der die Funktion „CSTOP" zugeordnet ist, um den zyklischen Betrieb aufzurufen. Im Feld „MODE" wird „CYCLE" angezeigt und die LED auf dem [END]-Schalter des Steuergeräts blinkt. Der Betrieb stoppt nach Ausführung der letzten Programmzeile oder des End-Befehls.



④ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE" zugeordnet ist, um zum RUN-Menü zurückzukehren.



HINWEIS

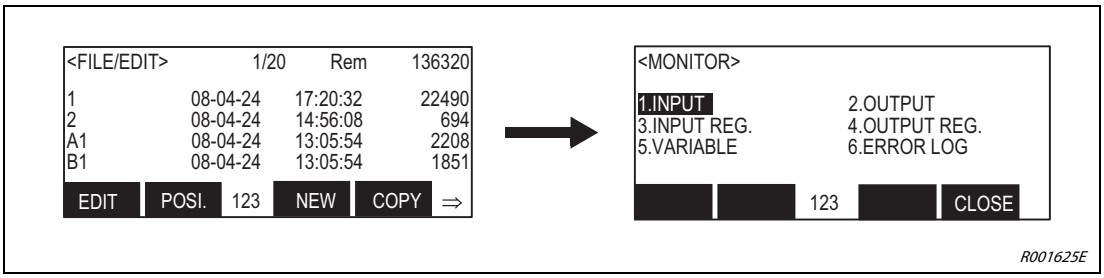
Wird die Programmausführung während des zyklischen Betriebs durch Betätigung der [STOP]-Taste gestoppt, erfolgt ein Wechsel in den kontinuierlichen Betrieb. Möchten Sie den Betrieb im zyklischen Modus fortsetzen, betätigen Sie die [F4]-Taste nach Betätigung der [START]-Taste. (Eine Umschaltung in den zyklischen Betrieb ist auch durch Betätigung des [END]-Schalters auf dem Steuergerät möglich.)

3.17 Monitor-Funktionen

Im Folgenden werden die aufgelisteten Monitor-Funktionen beschrieben:

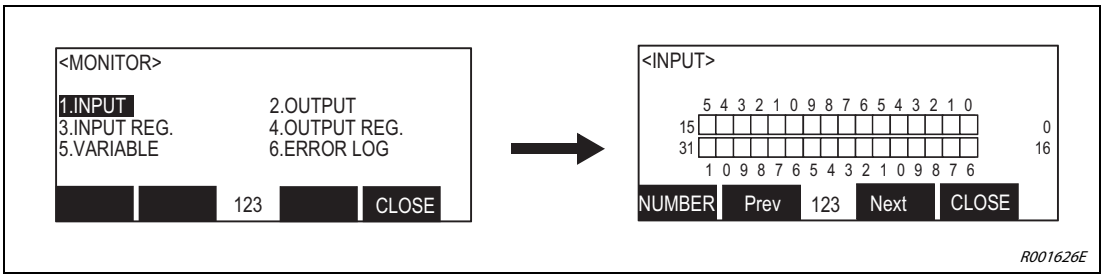
- Monitor-Funktion für Eingangssignale
- Monitor-Funktion für Ausgangssignale
- Monitor-Funktion für Eingangsregister
- Monitor-Funktion für Ausgangsregister
- Monitor-Funktion für Variable
- Liste der aufgetretenen Fehlermeldungen

Betätigen Sie zum Aufruf der oben genannten Funktionen die [MONITOR]-Taste der Teaching Box. Die Funktion ist auch ohne zugewiesene Betriebsrechte der Teaching Box (disable) aufrufbar. Die Monitor-Funktion für Variable kann nicht im Menü zur Programmeditierung aufgerufen werden.

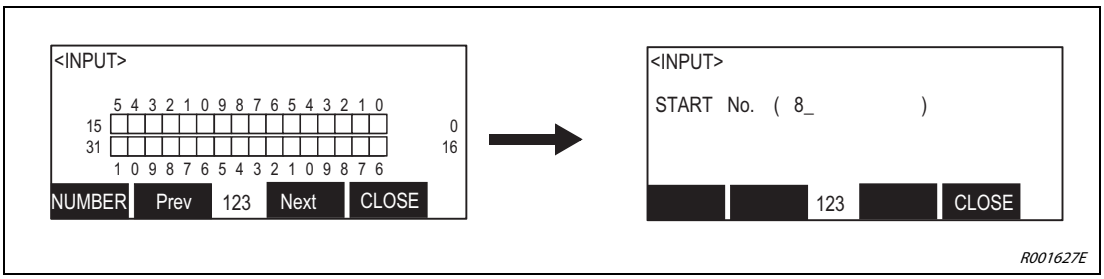


3.17.1 Monitor-Funktion für Eingangssignale

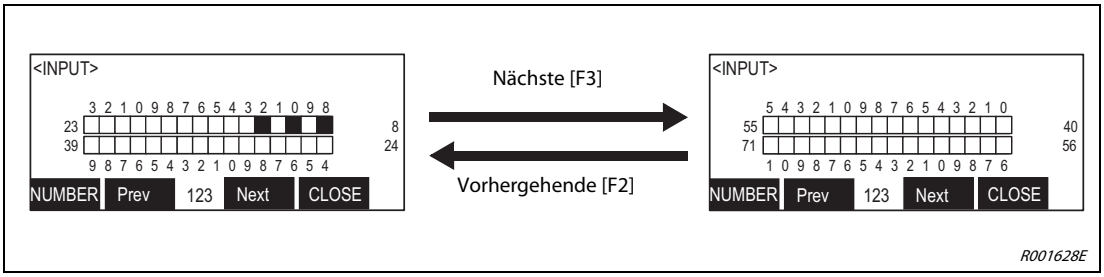
① Betätigen Sie die Taste [1] im Monitormenü, um die Monitor-Funktion für die Eingangssignale aufzurufen. Es werden 32 Signale auf einem Bildschirm angezeigt.


<MONITOR>
1.INPUT 2.OUTPUT
3.INPUT REG. 4.OUTPUT REG.
5.VARIABLE 6.ERROR LOG
123 CLOSE
<INPUT>
5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0
15 0
31 16
1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6
NUMBER Prev 123 Next CLOSE
R001626E

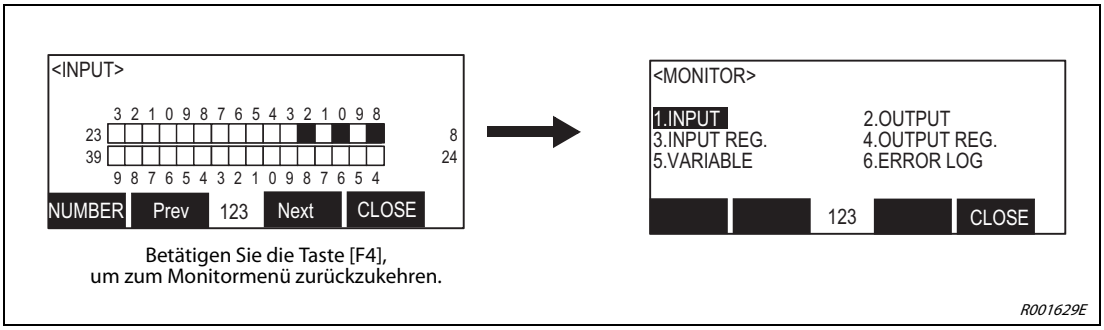
② Die Signale 8 bis 39 sollen angezeigt werden. Betätigen Sie die Funktionstaste, die der Funktion „NUMBER" zugeordnet ist, um die Startnummer „8" einzugeben.



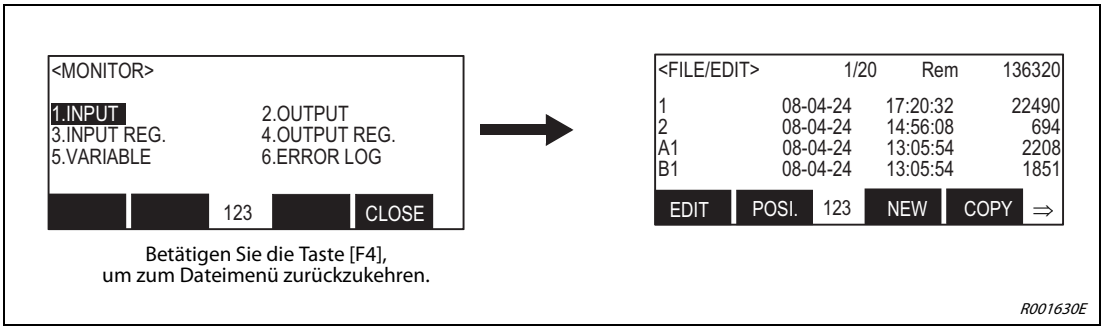
③ Der EIN/AUS-Zustand der 32 Signale 8 bis 31 wird angezeigt. Schwarz bedeutet EIN und Weiß bedeutet AUS.
Betätigen Sie die Taste [F3](Next), um die nächsten 32 Signale anzuzeigen und die Taste [F2](Prev), um die vorhergehenden Signale anzuzeigen.



④ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Monitormenü zurückzukehren.



⑤ Betätigen Sie im Monitormenü die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Dateimenü zurückzukehren.



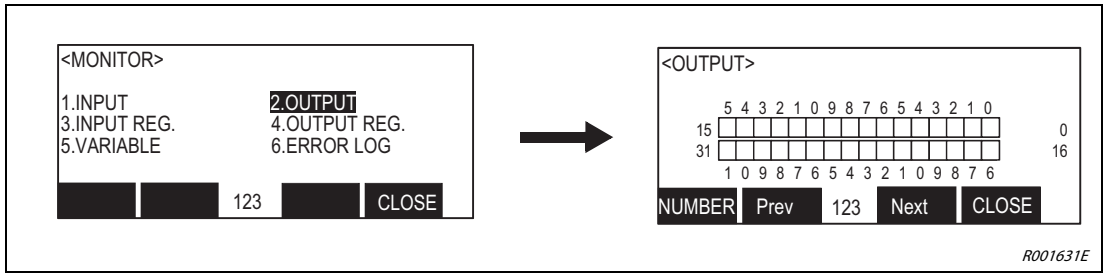
HINWEIS

Bei Betätigung der [MONITOR]-Taste im bereits geöffneten Monitormenü wird das Monitormenü beendet und das Dateimenü aufgerufen.

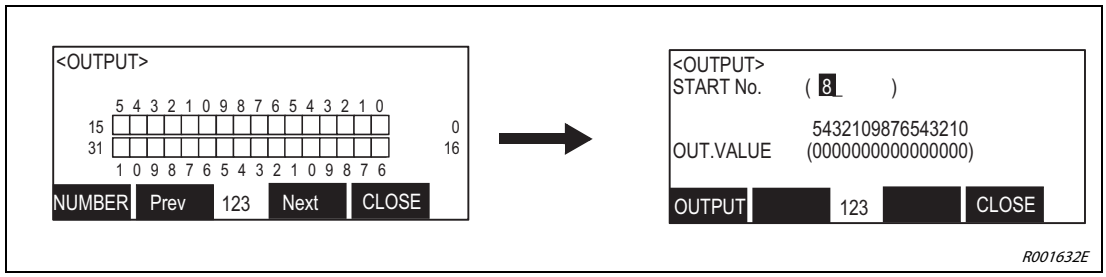
3.17.2 Monitor-Funktion für Ausgangssignale

Die Funktion ermöglicht die Anzeige und Einstellung der Ausgangssignalzustände.

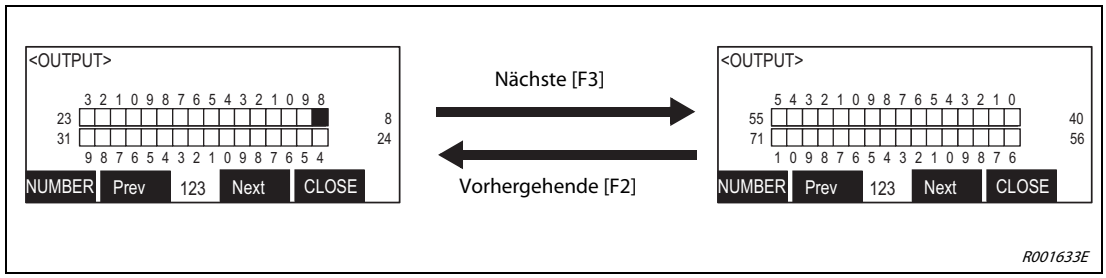
① Betätigen Sie die Taste [2] im Monitormenü, um die Monitor-Funktion für die Ausgangssignale aufzurufen. Es werden 32 Signale auf einem Bildschirm angezeigt.



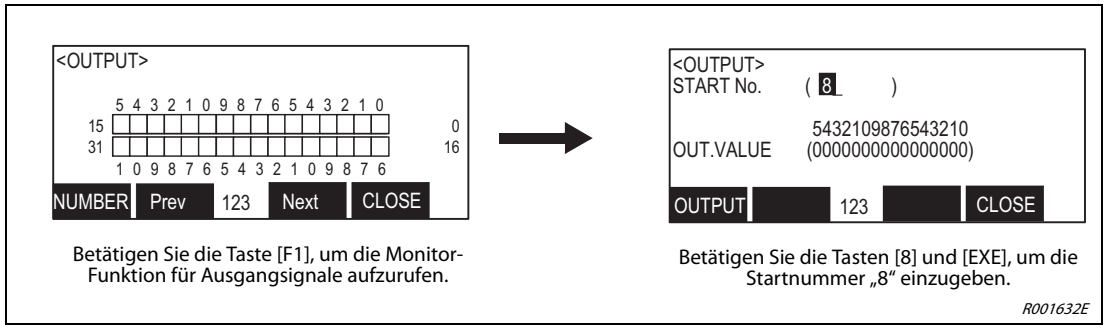
② Die Signale 8 bis 39 sollen angezeigt werden. Betätigen Sie die Funktionstaste, die der Funktion „NUMBER" zugeordnet ist, um die Startnummer „8" einzugeben.



Obwohl die Signalzustände der Ausgangssignale angezeigt werden, ist ein zwangsweises Schalten der Signale aus diesem Menü heraus nicht möglich.
Betätigen Sie die Funktionstaste, die der Funktion „CLOSE" zugeordnet ist. Der EIN/AUS-Zustand der 32 Signale 8 bis 31 wird angezeigt. Schwarz bedeutet EIN und Weiß bedeutet AUS.
Betätigen Sie die Taste [F3](Next), um die nächsten 32 Signale anzuzeigen und die Taste [F2](Prev), um die vorhergehenden Signale anzuzeigen.



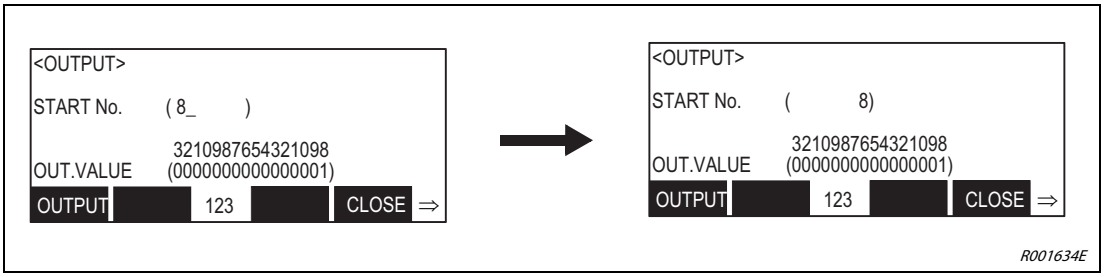
③ Erzwungenes Ausgangssignal
Das Ausgangssignal 8 soll ausgeschaltet werden. Betätigen Sie die Funktionstaste, die der Funktion „NUMBER" zugeordnet ist. Geben Sie die Startnummer „8" ein.



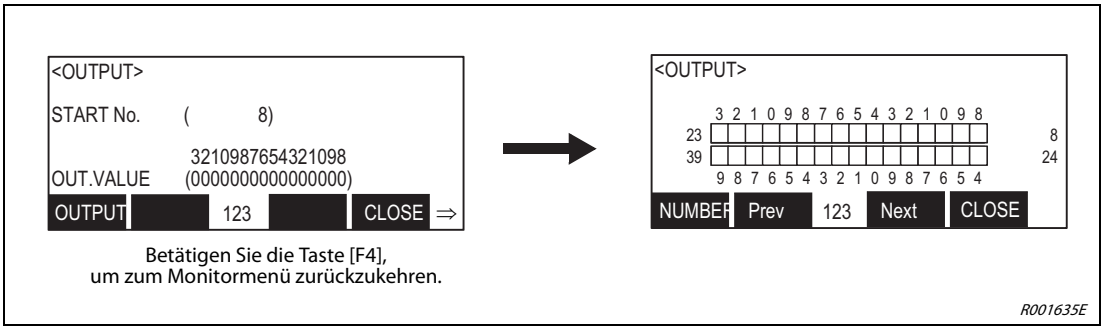
Betätigen Sie die Taste [F1], um die Monitor-Funktion für Ausgangsignale aufzurufen.

Betätigen Sie die Tasten [8] und [EXE], um die Startnummer „8" einzugeben.

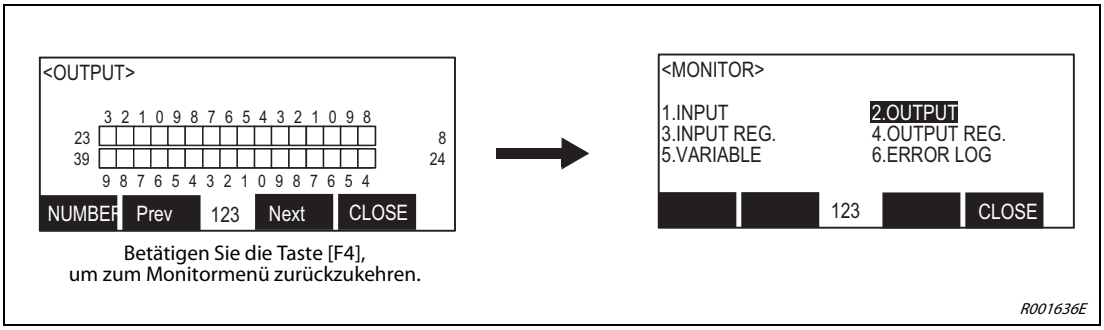
④ Bewegen Sie den Cursor mit den Pfeiltasten zum Wert für die 8. Da das Signal eingeschaltet ist, wird eine „1“ angezeigt. Geben Sie den Wert „0“ ein und betätigen Sie die Funktionstaste, die der Funktion „OUTPUT“ zugeordnet ist, um den Ausgang auszuschalten.



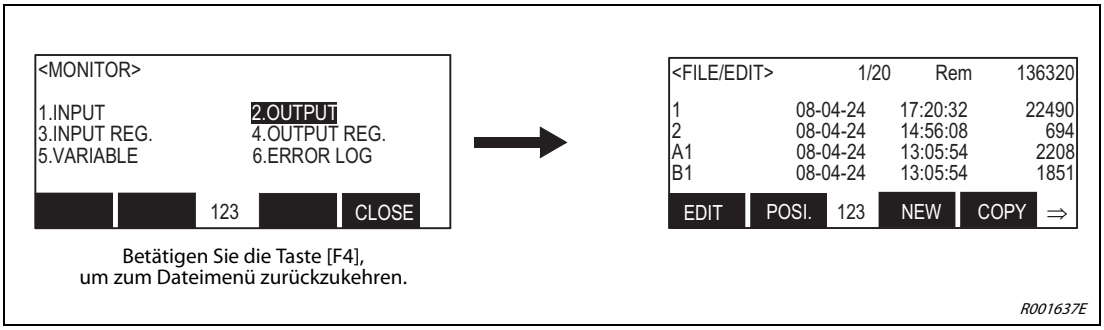
⑤ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um die Monitor-Funktion für die Ausgangssignale aufzurufen.


<OUTPUT>
START No. (8)
3210987654321098
OUT.VALUE (0000000000000000)
OUTPUT 123 CLOSE ⇒
Betätigen Sie die Taste [F4],
um zum Monitormenü zurückzukehren.
<OUTPUT>
3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8
23 8
39 24
9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4
NUMBER Prev 123 Next CLOSE
R001635E

⑥ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Monitormenü zurückzukehren.


<OUTPUT>
3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8
23 8
39 24
9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4
NUMBER Prev 123 Next CLOSE
Betätigen Sie die Taste [F4],
um zum Monitormenü zurückzukehren.
<MONITOR>
1.INPUT 2.OUTPUT
3.INPUT REG. 4.OUTPUT REG.
5.VARIABLE 6.ERROR LOG
123 CLOSE
R001636E

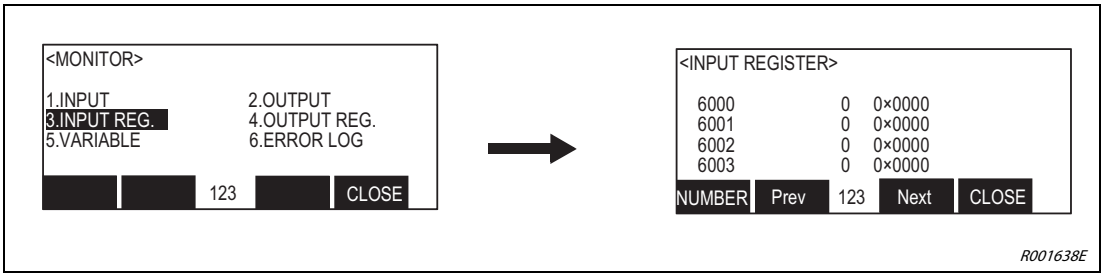
⑦ Betätigen Sie im Monitormenü die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Dateimenü zurückzukehren.



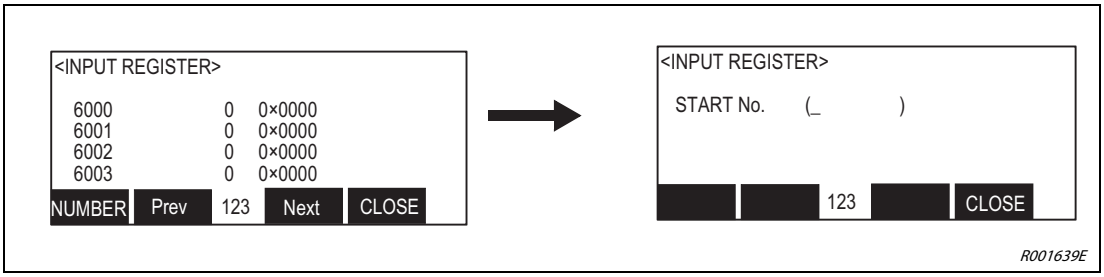
3.17.3 Monitor-Funktion für Eingangsregister

Mit dieser Funktion können die Eingangsregister für den CC-Link-Betrieb angezeigt werden. (Bei den Steuergeräten CRnQ kann die Funktion nicht verwendet werden.)

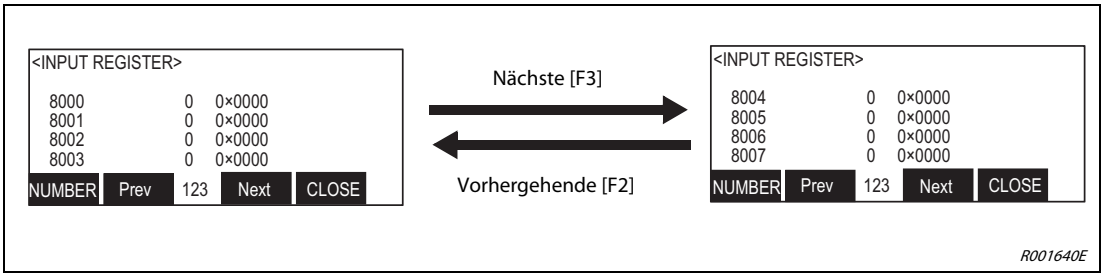
① Betätigen Sie die Taste [3] im Monitormenü, um die Monitor-Funktion für Eingangsregister aufzurufen. Es werden 4 Register auf einem Bildschirm angezeigt.



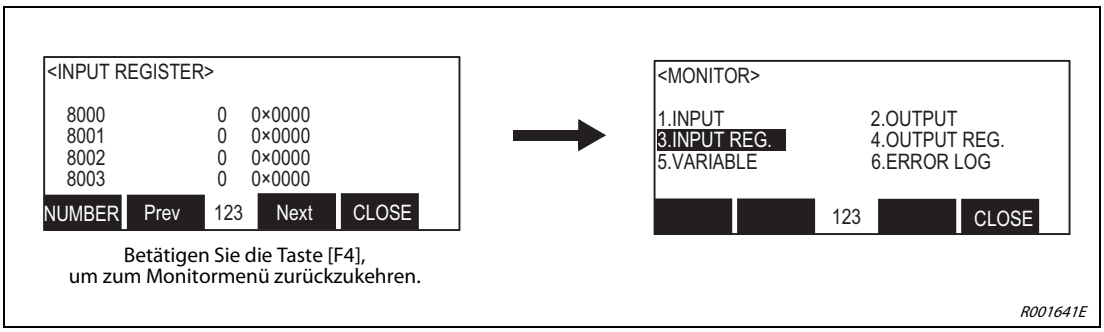
② Der Status des Eingangsregisters 8000 soll angezeigt werden. Betätigen Sie die Funktionstaste, die der Funktion „NUMBER" zugeordnet ist, und geben Sie die Startnummer „8000" ein.



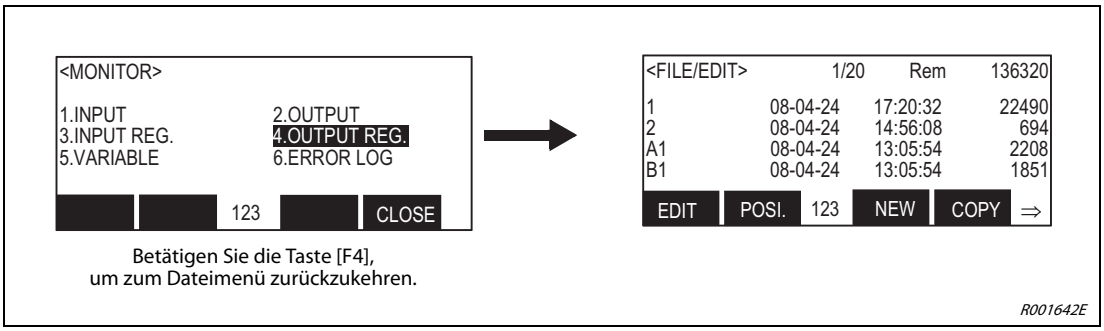
③ Der Status der vier Eingangsregisters ab Register 8000 wird angezeigt. Betätigen Sie die Taste [F3](Next), um die nächsten 4 Register anzuzeigen und die Taste [F2](Prev), um die vorhergehenden Register anzuzeigen.



④ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE" zugeordnet ist, um zum Monitormenü zurückzukehren.



⑤ Betätigen Sie im Monitormenü die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Dateimenü zurückzukehren.



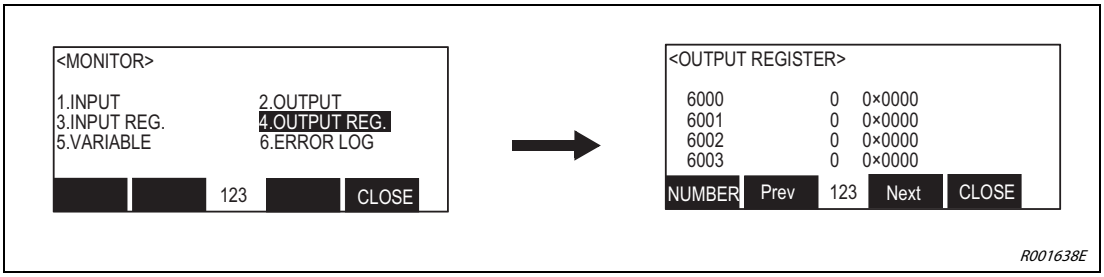
HINWEIS

Bei Betätigung der [MONITOR]-Taste im bereits geöffneten Monitormenü wird das Monitormenü beendet und das Dateimenü aufgerufen.

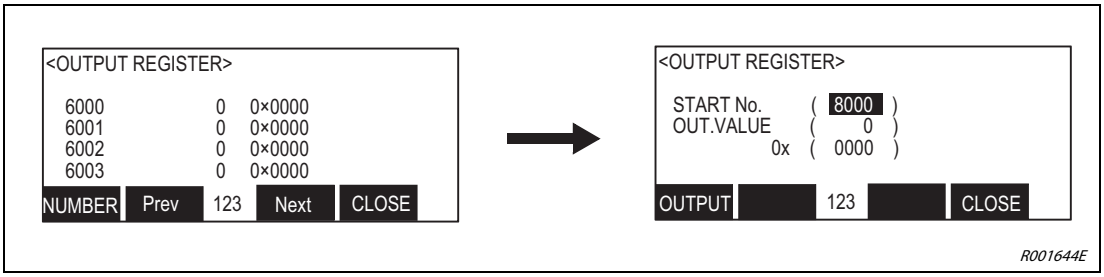
3.17.4 Monitor-Funktion für Ausgangsregister

Mit dieser Funktion können die Ausgangsregister für den CC-Link-Betrieb angezeigt werden. (Bei den Steuergeräten CRnQ kann die Funktion nicht verwendet werden.)

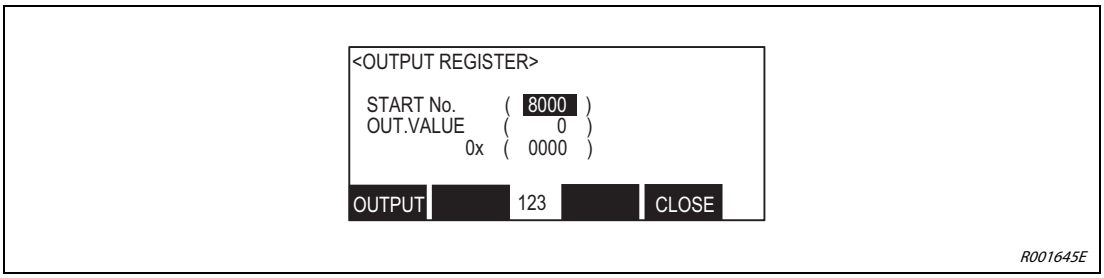
① Betätigen Sie die Taste [4] im Monitormenü, um die Monitor-Funktion für Ausgangsregister aufzurufen. Es werden 4 Register auf einem Bildschirm angezeigt.



② Der Status des Ausgangsregisters 8000 soll angezeigt werden. Betätigen Sie die Funktionstaste, die der Funktion „NUMBER“ zugeordnet ist, und geben Sie die Startnummer „8000“ ein.

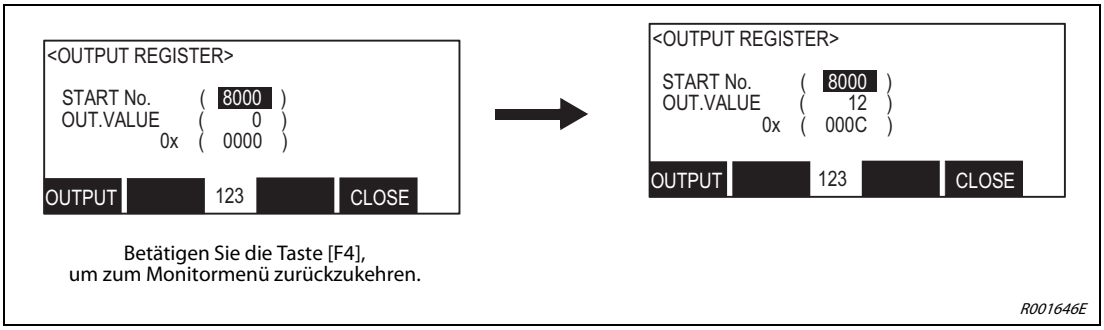


③ Der Status des Ausgangsregisters 8000 wird in Klammern in der Zeile „OUT.VALUE“ angezeigt. Darunter erscheint in der Zeile 0x der hexadezimale Wert.

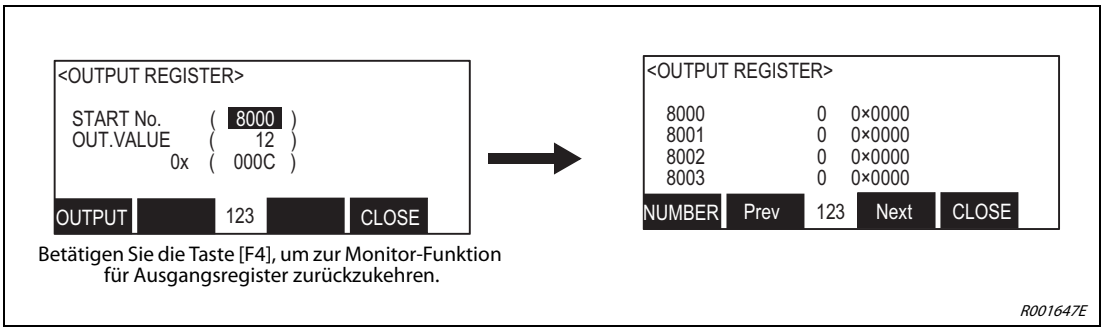


Wenn Sie die Funktionstaste betätigen, die der Funktion „CLOSE“ zugeordnet ist, wird wieder das Monitormenü mit der Anzeige des Registers 8000 aufgerufen. Der Inhalt des Registers kann im aktuellen Menü geändert werden. Im Folgenden wird der Inhalt des Registers 8000 auf dezimal 12 gesetzt.

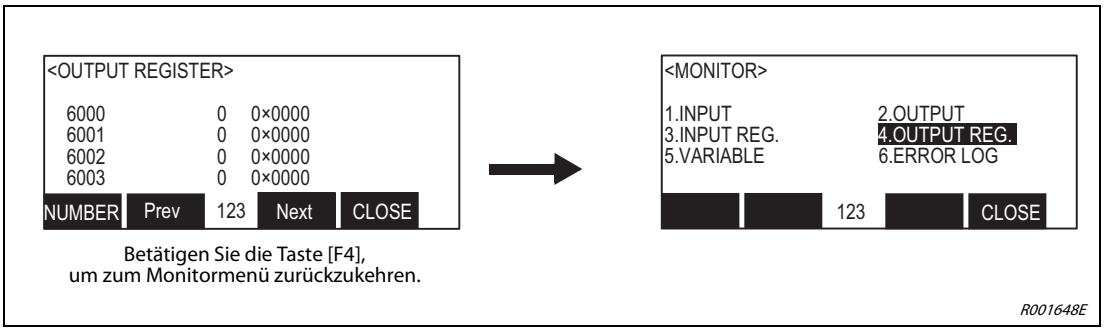
④ Der Inhalt des Registers kann dezimal oder hexadezimal eingestellt werden. Bewegen Sie den Cursor mit den Pfeiltasten zum Feld „OUT. VALUE“, um die Einstellung dezimal vorzunehmen und geben Sie den Wert „12“ ein. Sie können Zeichen mit der [CLEAR]-Taste löschen.
Bewegen Sie den Cursor zum Feld „0x“, um die Einstellung hexadezimal vorzunehmen. Geben Sie hier das Zeichen „C“ ein. Sie können Zeichen mit der [CLEAR]-Taste löschen.
Betätigen Sie die Funktionstaste, die der Funktion „OUTPUT“ zugeordnet ist, um die aktuelle Eingabe zu übernehmen.



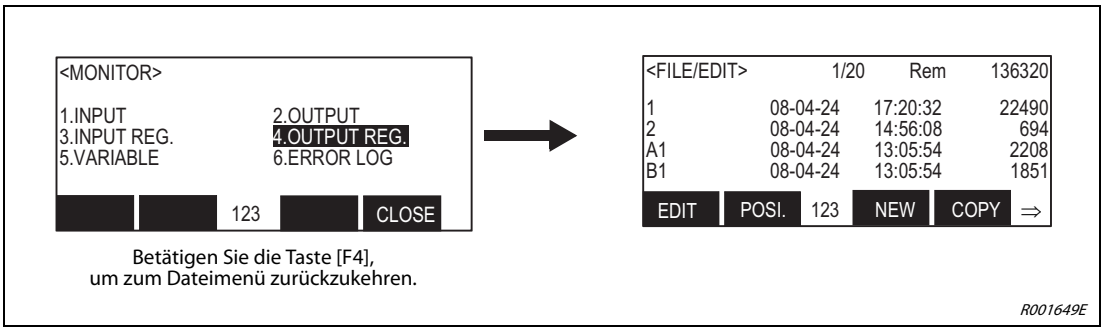
⑤ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zur Monitor-Funktion für Ausgangsregister zurückzukehren.



⑥ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zur Monitor-Funktion für Ausgangssignale zurückzukehren.



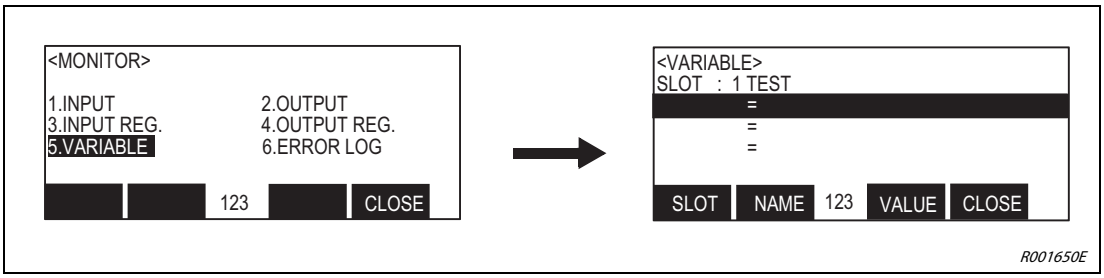
⑦ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Dateimenü zurückzukehren.



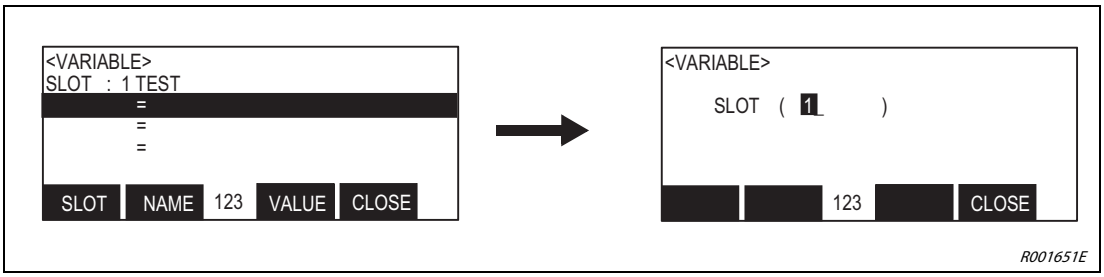
3.17.5 Monitor-Funktion für Variable

Die Funktion ermöglicht die Anzeige und Einstellung von Variablen.

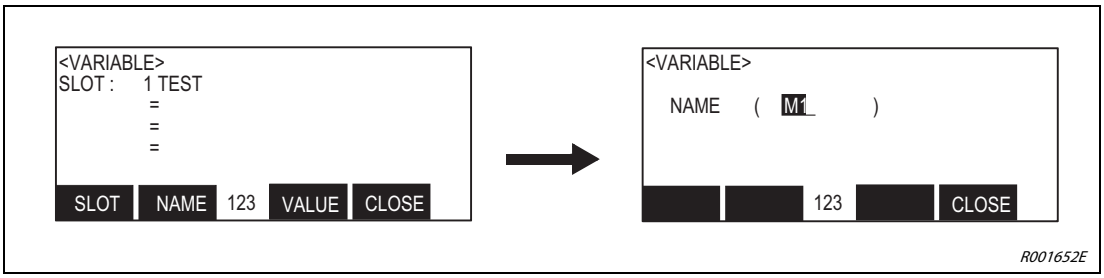
① Betätigen Sie die Taste [5] im Monitormenü, um die Monitor-Funktion für Variable aufzurufen.



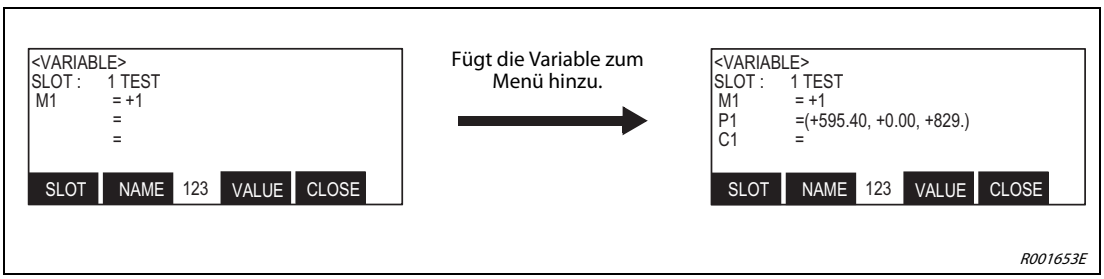
② Wählen Sie das Zielprogramm und den Programmplatz.
Betätigen Sie die Funktionstaste, die der Funktion „SLOT“ zugeordnet ist, um den Programmplatz auszuwählen. Wird kein Multitasking verwendet, ist die Progammplatznummer „1“.



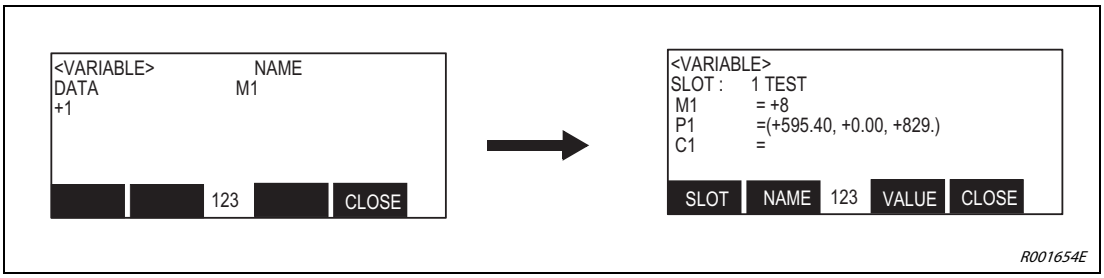
③ Die Programmplatznummer und der Programmname werden in der Zeile „SLOT“ angezeigt. Betätigen Sie die Funktionstaste, die der Funktion „NAME“ zugeordnet ist, um den Variablennamen einzugeben.



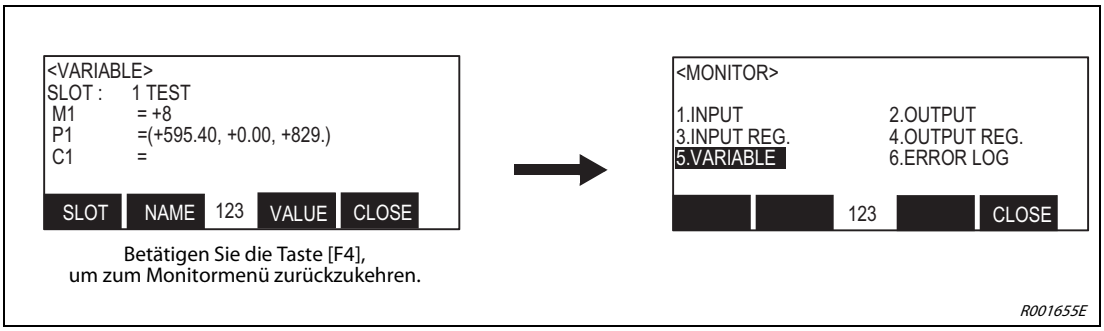
④ Anzeige des Variablenwerts
Sie können bis zu drei Variablen anzeigen, wenn Sie den Cursor in eine leere Zeile platzieren und den oben genannten Schritt wiederholen.



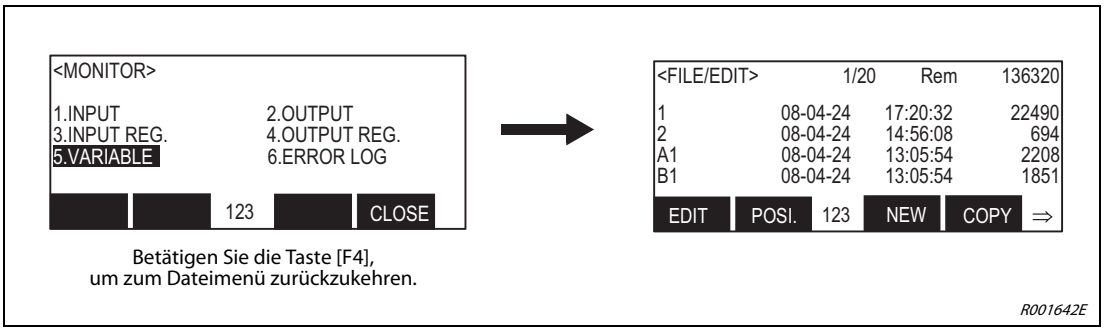
⑤ Einstellung des Variablenwerts
Der Wert der aktuell angezeigten Variablen kann geändert werden. Bewegen Sie den Cursor mit den Pfeiltasten zu der Variablen, die Sie ändern möchten, und betätigen Sie die Funktionstaste, die der Funktion „VALUE“ zugeordnet ist. Der aktuelle Wert wird angezeigt und kann geändert werden.



⑥ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Monitormenü zurückzukehren.



⑦ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Dateimenü zurückzukehren.



HINWEISE

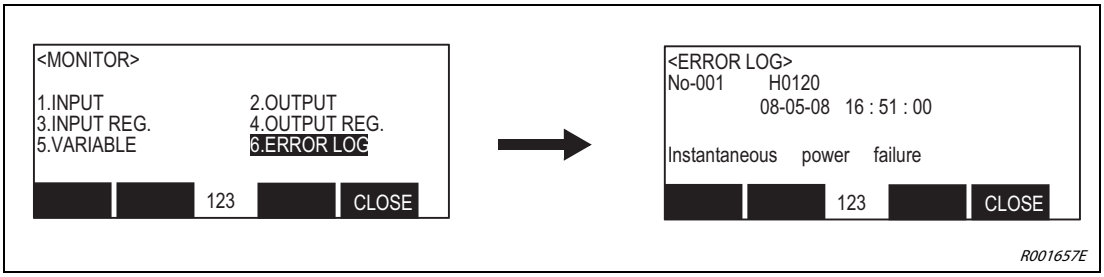
Die Anzeige von Variablen kann auch ohne zugewiesene Betriebsrechte der Teaching Box erfolgen. Eine Änderung der Variablen ist auch im Automatikbetrieb möglich.

Bei Betätigung der [MONITOR]-Taste wird das Monitormenü beendet und das Dateimenü aufgerufen.

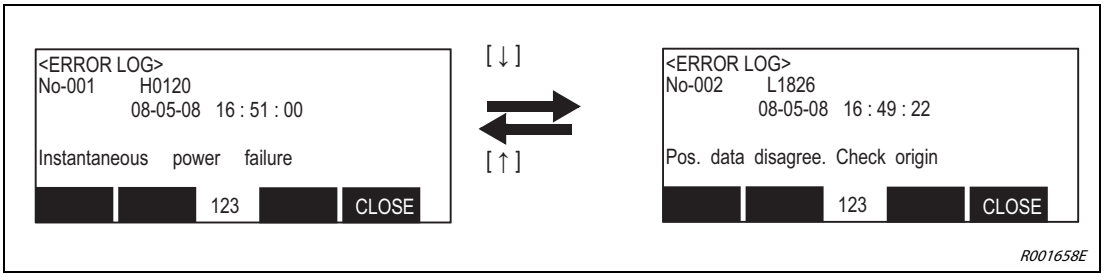
3.17.6 Liste der aufgetretenen Fehlermeldungen

Die bisher aufgetretenen Fehlermeldungen werden in einer Liste angezeigt. Diese Funktion ist für eine Störungssuche sehr hilfreich.

① Betätigen Sie die Taste [6] im Monitormenü, um die Liste der aufgetretenen Fehlermeldungen aufzurufen.



② Zeigen Sie den gewünschten Fehler mit Hilfe der Pfeiltasten an.



HINWEIS Die Anzeige von Fehlermeldungen kann auch ohne zugewiesene Betriebsrechte der Teaching Box erfolgen.

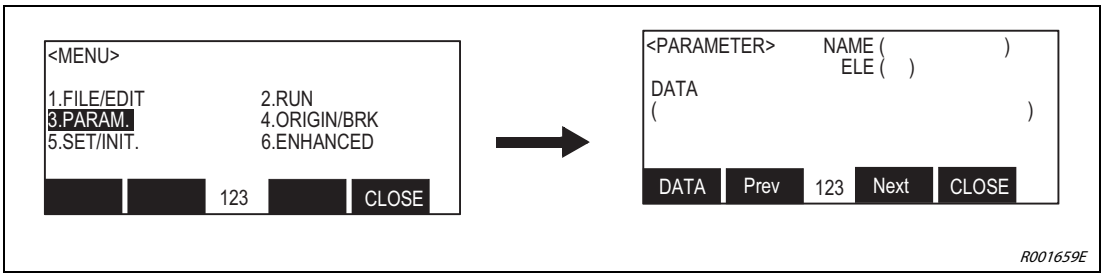
3.18 Parametermenü

3.18.1 Parameter einstellen

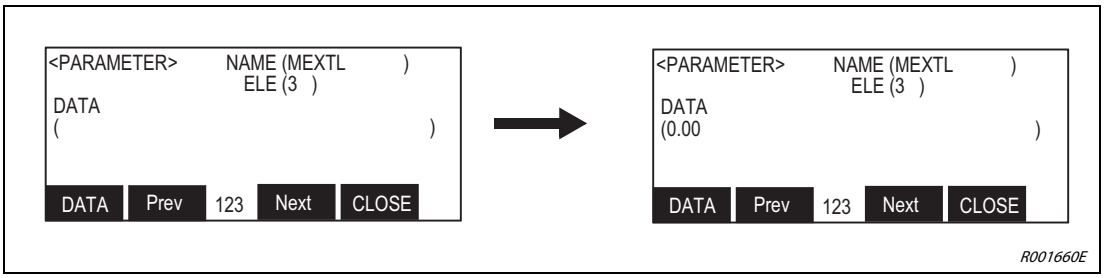
Die Funktionen der parallelen Ein-/Ausgabeschnittstelle, die Werkzeuglänge usw. sind über Parameter festgelegt. Diese Parameter können im Teach-Modus eingestellt werden.

Beispiel▽ Im folgenden Beispiel wird der Wert für die Z-Achse des Parameters „MEXTL (Werkzeugdaten)“ von 0 auf 100 mm gesetzt.

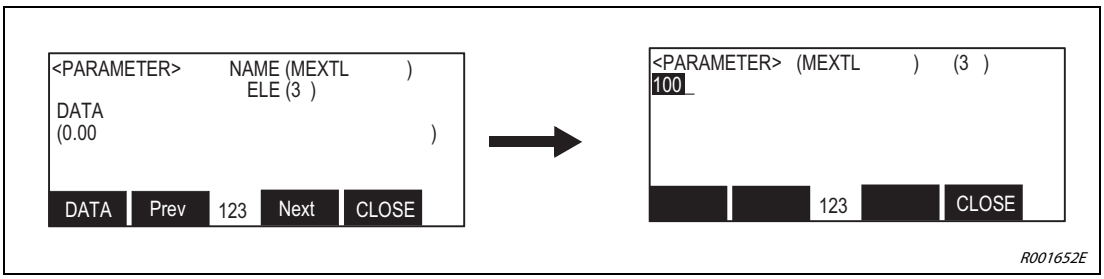
① Betätigen Sie die Taste [3] im Hauptmenü, um das Menü zur Einstellung von Parametern aufzurufen.



② Geben Sie im Feld „NAME“ den Parameternamen „MEXTL“ und im Feld „ELE“ den Wert „3“ ein. Die aktuellen Daten werden angezeigt.

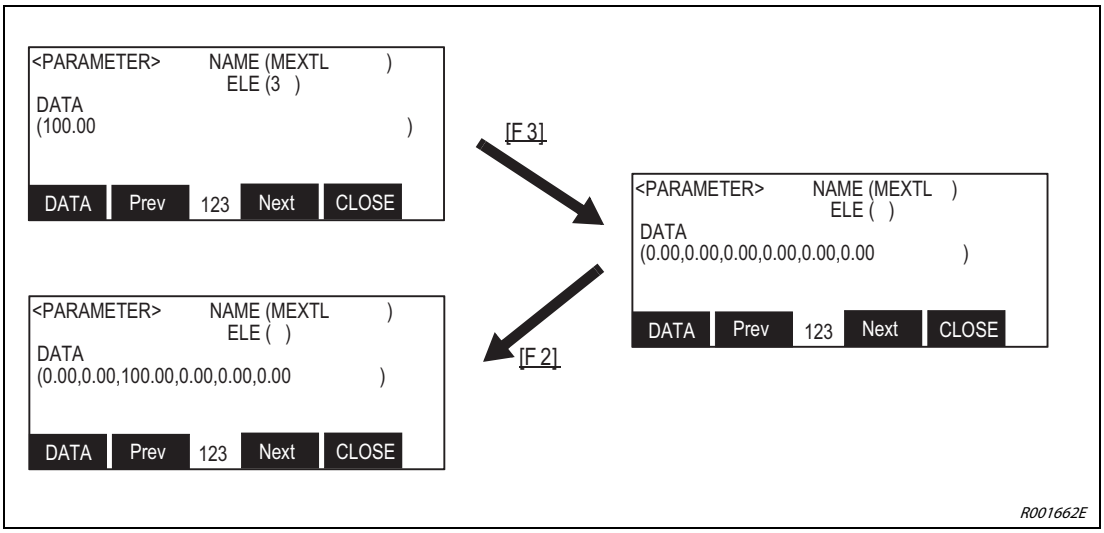


③ Betätigen Sie die Funktionstaste, die der Funktion „DATA“ zugeordnet ist, und geben Sie den neuen Wert „100“ ein. Mit Hilfe der [CLEAR]-Taste können nicht benötigte Daten gelöscht werden.

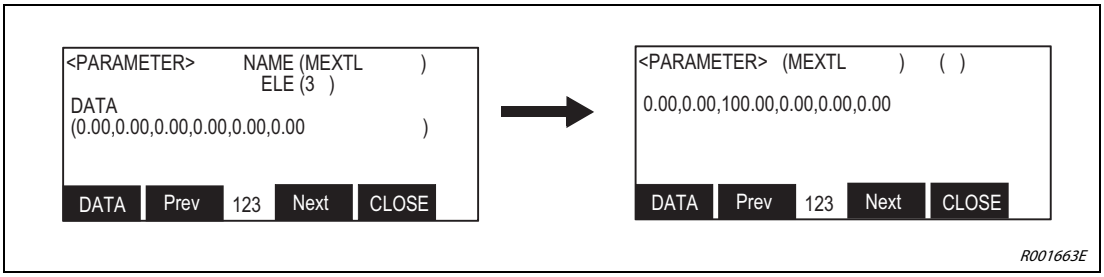


Bei Betätigung der [EXE]-Taste ertönt ein Signalton, der Wert wird gespeichert und die Anzeige des Menüs zur Einstellung von Parametern erscheint.
Auch nach Eingabe des neuen Wertes kann der Vorgang durch Betätigung der Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, abgebrochen werden und die Anzeige des Menüs zur Einstellung von Parametern erscheint.

Betätigen Sie die Funktionstaste, die der Funktion „Next“ zugeordnet ist, um den nächsten Parameter aufzurufen. Betätigen Sie die Funktionstaste, die der Funktion „Prev“ zugeordnet ist, um den vorhergehenden Parameter aufzurufen. In diesen Fällen werden alle Elemente des Parameters angezeigt.



Der Wert des Parameters kann auch in diesen Anzeigen verändert werden.
Betätigen Sie dazu die Funktionstaste, die der Funktion „DATA“ zugeordnet ist. Bewegen Sie den Cursor mit Hilfe der Pfeiltaste zu dem Element, das Sie ändern möchten. Geben Sie den neuen Wert ein. Nicht benötigte Daten können mit der [CLEAR]-Taste gelöscht werden.



Bei Betätigung der [EXE]-Taste ertönt ein Signalton, der Wert wird gespeichert und die Anzeige des Menüs zur Einstellung von Parametern erscheint.
Auch nach Eingabe des neuen Wertes kann der Vorgang durch Betätigung der Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, abgebrochen werden und die Anzeige des Menüs zur Einstellung von Parametern erscheint.

△

HINWEISE

Damit ein neuer Parameterwert wirksam wird, muss die Spannungsversorgung des Steuergerätes aus- und wieder eingeschaltet werden.

Während der Ausführung des Programms ist keine Einstellung von Parametern möglich. Wird versucht, während der Ausführung des Programms einen Parameter zu ändern, erfolgt eine Fehlermeldung. Das Programm wird trotz Ausgabe der Fehlermeldung nicht unterbrochen.

Auch wenn bei der Eingabe des Parameternamens nicht alle Zeichen korrekt eingegeben werden, erfolgt automatisch die Anzeige ähnlicher Parameternamen.

3.19 Gelenkbremsen lösen

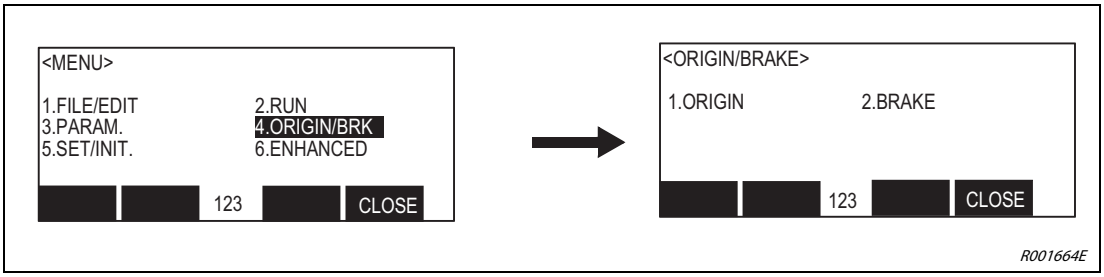
Die Bremsen für die Robotergelenke können bei ausgeschalteter Servospannung gelöst werden. Dies ermöglicht eine direkte manuelle Bewegung des Roboterarms.

ACHTUNG:

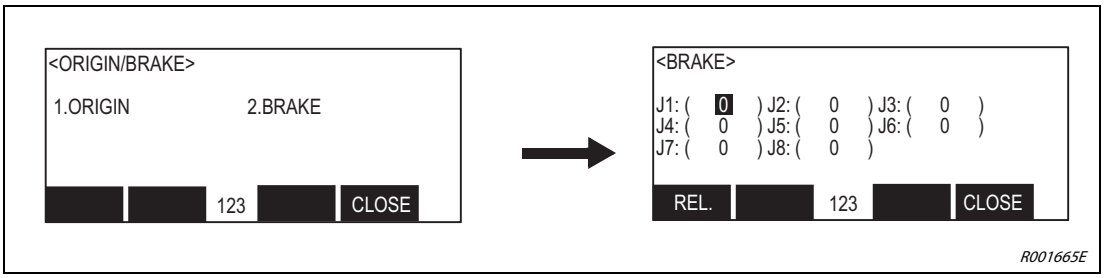
Beachten Sie, dass der Roboterarm aufgrund des Eigengewichts bei gelösten Bremsen heruntersinken kann. Unterstützen Sie daher den Roboterarm vor dem Lösen der Bremsen.

Schalten Sie zuerst die Servos aus (siehe Abschn. 3.11). Bringen Sie anschließend den Dreistufenschalter in Mittelstellung und führen Sie die nachfolgenden Schritte aus.

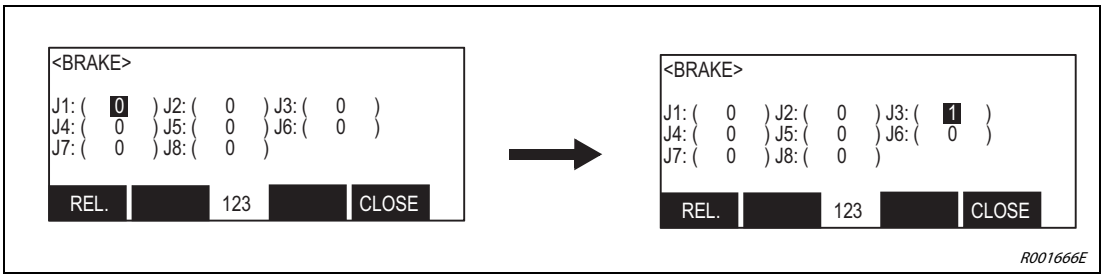
① Betätigen Sie die Taste [4] im Hauptmenü, um das Menü zur Einstellung der Grundposition und zum Lösen der Gelenkbremsen aufzurufen.



② Betätigen Sie die Taste [2], um das Menü zum Lösen der Gelenkbremsen aufzurufen.



③ Setzen Sie die Achse, deren Bremse gelöst werden soll, auf „1“.

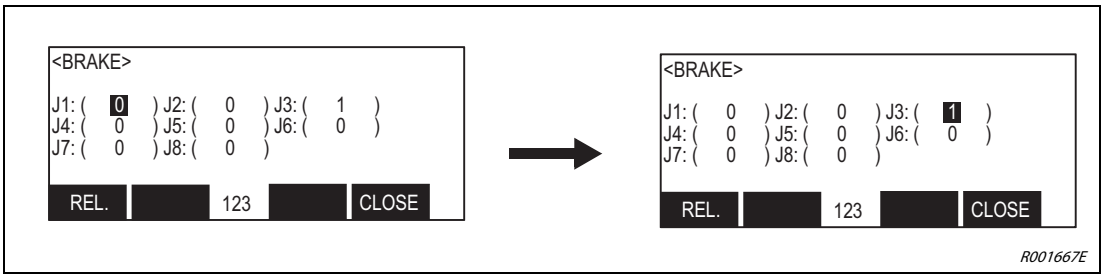


④ Betätigen Sie die Funktionstaste, die der Funktion „REL.“ (Release) zugeordnet ist, und halten Sie diese gedrückt, um die Bremse zu lösen.

Knickarmroboter (RV-3SD usw.): Die Bremse der festgelegten Achse ist gelöst, solange Sie die Funktionstaste F1 betätigen.

SCARA-Roboter (RH-20SDH usw.): Die Bremse der Hubachse wird intermittierend gelöst (d.h. Bremse wird gelöst, Bremse greift, Bremse wird gelöst usw.), solange Sie die Funktionstaste F1 betätigen.

Wird die Taste losgelassen oder der Dreistufenschalter losgelassen oder durchgedrückt, greift die Bremse wieder.

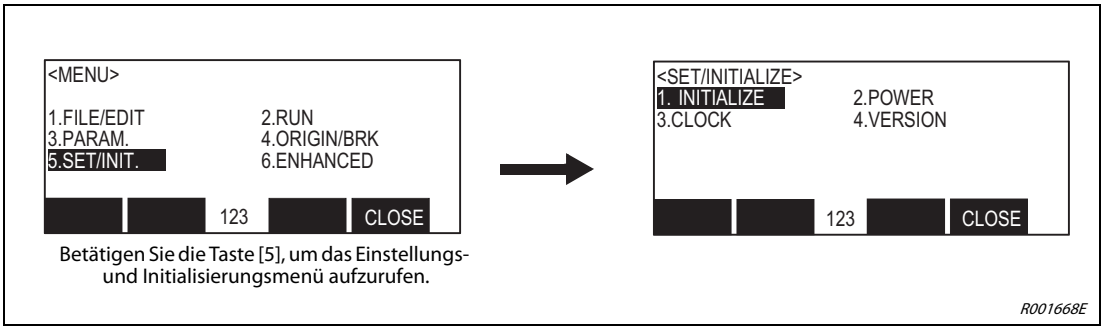


3.20 Einstellungs- und Initialisierungsmenü

Im Folgenden werden die aufgelisteten Monitor-Funktionen beschrieben:

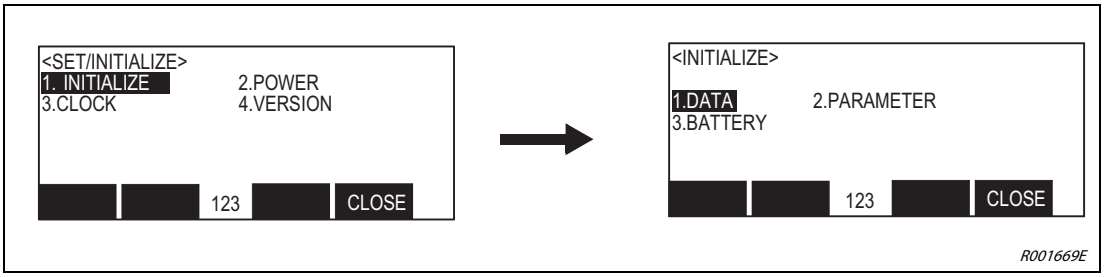
- Alle gespeicherten Programme löschen
- Alle Parameter auf die Werkseinstellung zurücksetzen
- Batteriezähler zurücksetzen
- Verbleibende Betriebszeit der Batterie und Einschaltzeit anzeigen
- Datum und Uhrzeit einstellen
- Software-Version der Teaching Box und des Steuergeräts anzeigen

Betätigen Sie die Taste [5] im Hauptmenü, um das Einstellungs- und Initialisierungsmenü aufzurufen.

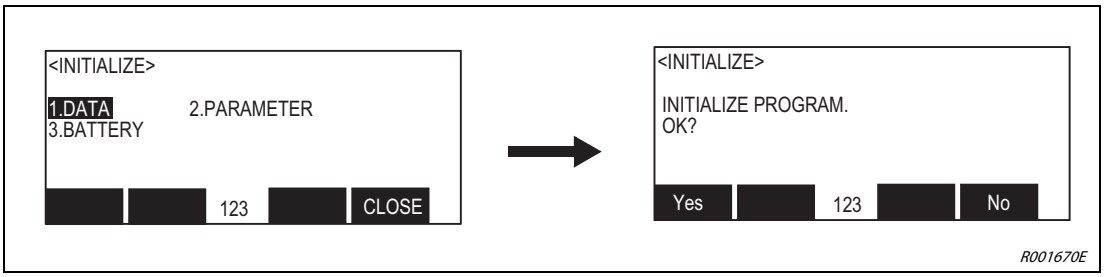


3.20.1 Alle gespeicherten Programme löschen

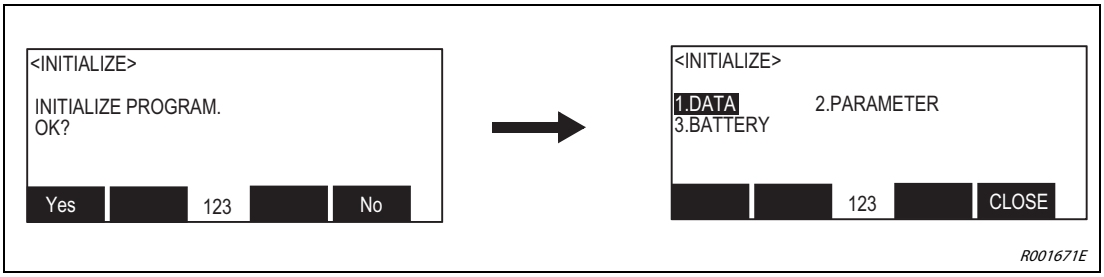
① Betätigen Sie die Taste [1] im Einstellungs- und Initialisierungsmenü, um das Initialisierungsmenü aufzurufen.



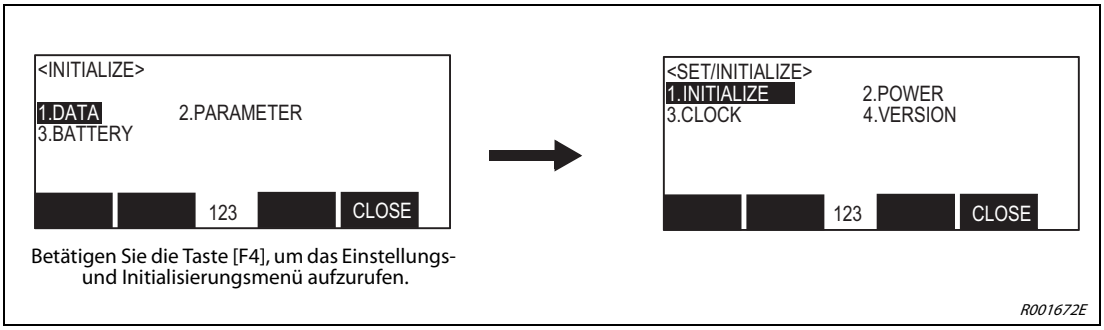
② Betätigen Sie die Taste [1], um das Menü zum Löschen aller gespeicherten Programme aufzurufen.



③ Betätigen Sie die Funktionstaste, die der Funktion „Yes“ zugewiesen ist, um alle Programme zu löschen. Betätigen Sie die Funktionstaste, die der Funktion „No“ zugewiesen ist, um den Vorgang abzubrechen. Das Initialisierungsmenü erscheint.



④ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Einstellungs- und Initialisierungsmenü zurückzukehren.

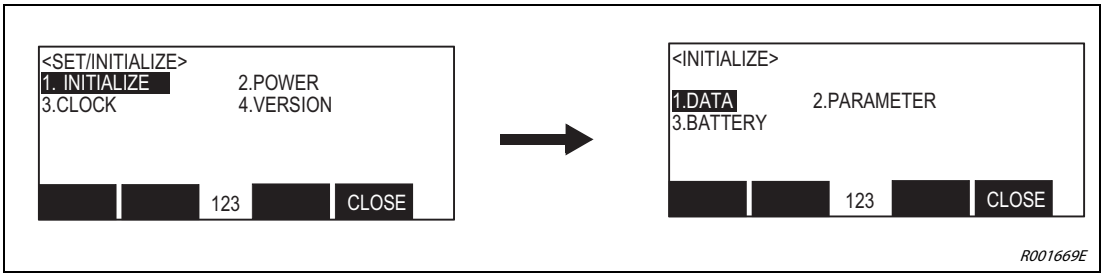


HINWEIS Es werden auch geschützte Programme und Programme mit geschützten Variablen gelöscht.

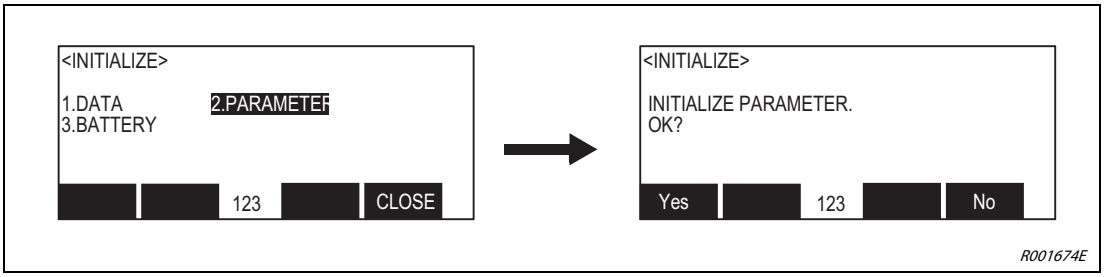
3.20.2 Alle Parameter auf die Werkseinstellung zurücksetzen

Alle Parameter werden auf die Werkseinstellung zurückgesetzt.

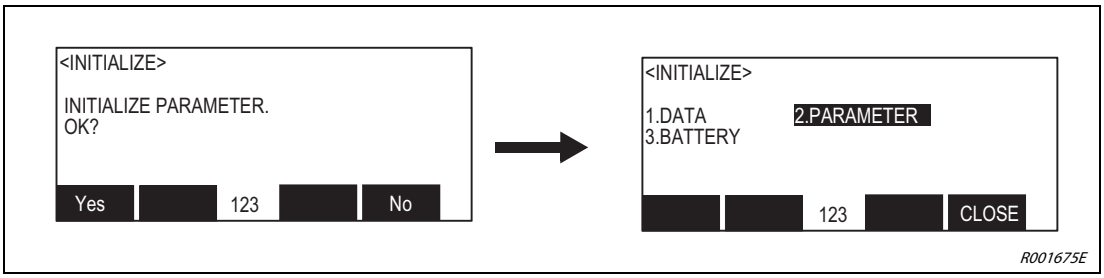
① Betätigen Sie die Taste [1] im Einstellungs- und Initialisierungsmenü, um das Initialisierungsmenü aufzurufen.



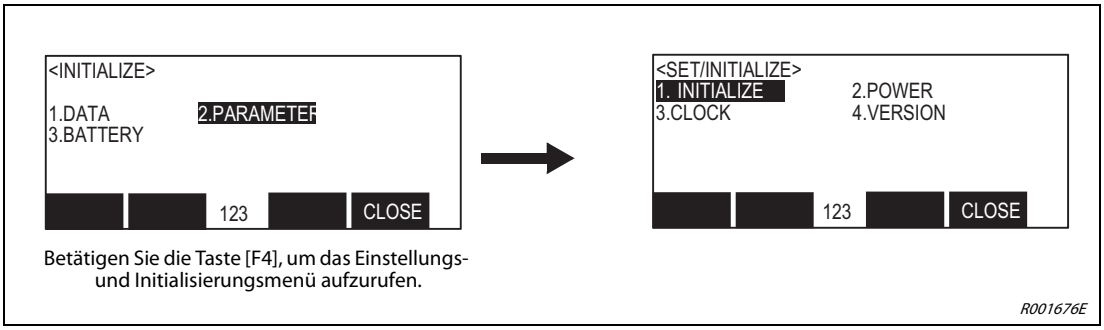
② Betätigen Sie die Taste [2], um das Menü zum Zurücksetzen aller Parameter aufzurufen.



③ Betätigen Sie die Funktionstaste, die der Funktion „Yes" zugewiesen ist, um alle Parameter auf die Werkseinstellung zurückzusetzen. Betätigen Sie die Funktionstaste, die der Funktion „No" zugewiesen ist, um den Vorgang abzubrechen. Das Initialisierungsmenü erscheint.



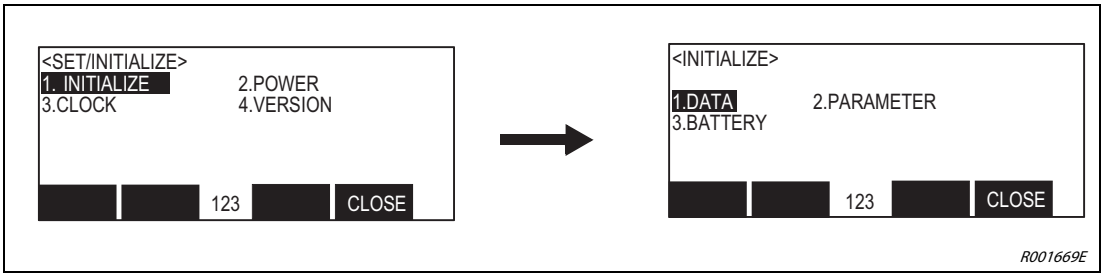
④ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE" zugeordnet ist, um zum Einstellungs- und Initialisierungsmenü zurückzukehren.



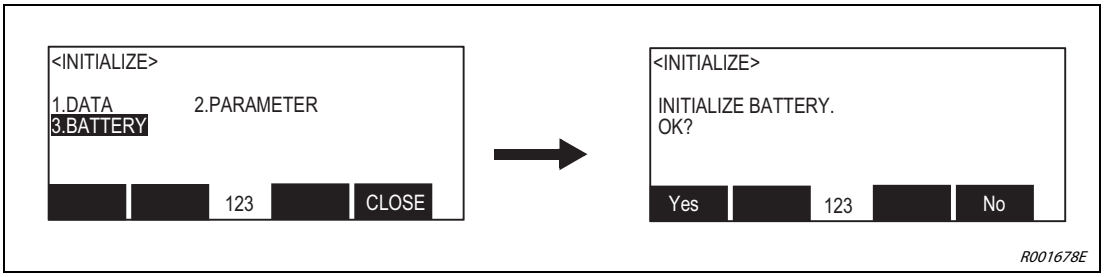
3.20.3 Batteriezähler zurücksetzen

Der Batteriezähler erfasst die Betriebszeit der Batterien im Roboterarm und im Steuergerät und dient als Referenz für die Warnmeldung zum Austausch der Batterien. Setzen Sie daher nach dem Austauschen der Batterien unbedingt den Batteriezähler zurück.

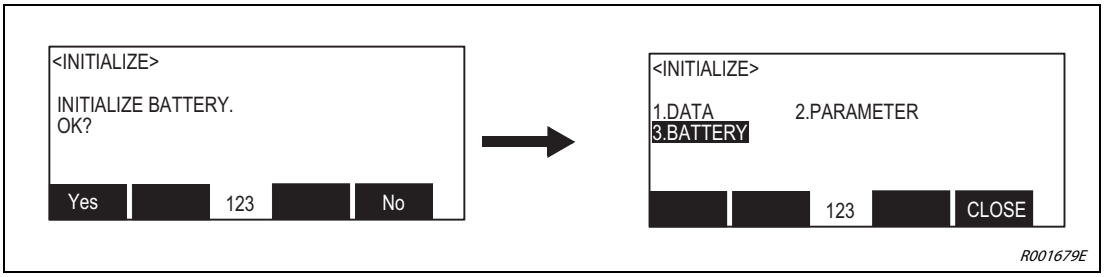
① Betätigen Sie die Taste [1] im Einstellungs- und Initialisierungsmenü, um das Initialisierungsmenü aufzurufen.



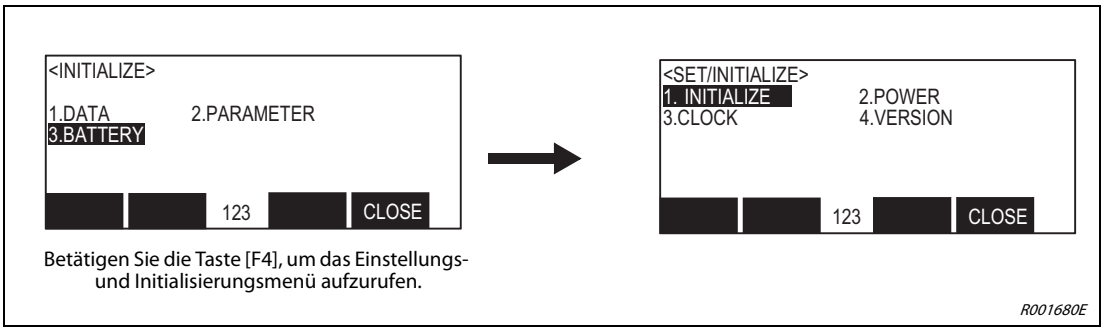
② Betätigen Sie die Taste [3], um das Menü zum Zurücksetzen des Batteriezählers aufzurufen.



③ Betätigen Sie die Funktionstaste, die der Funktion „Yes“ zugewiesen ist, um den Batteriezähler zurückzusetzen. Betätigen Sie die Funktionstaste, die der Funktion „No“ zugewiesen ist, um den Vorgang abzubrechen. Das Initialisierungsmenü erscheint.



④ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Einstellungs- und Initialisierungsmenü zurückzukehren.



HINWEISE

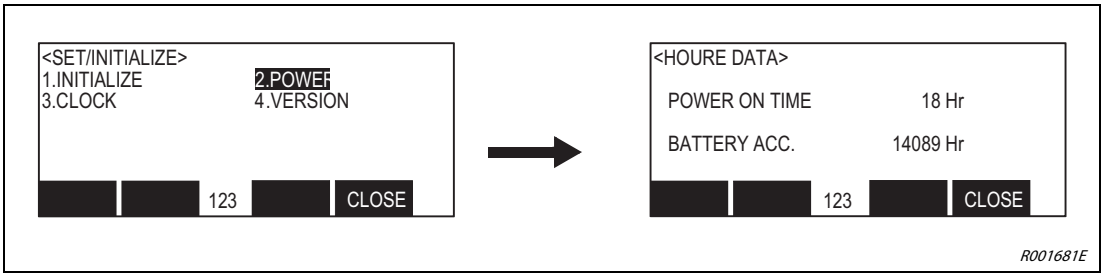
Der Batteriezähler zählt rückwärts. Gezählt wird nur im ausgeschalteten Zustand.

Bei verbrauchter Batterie wird eine Warnmeldung ausgegeben. Die Gebrauchszeit der Batterie wird ab dem Zurücksetzen des Batteriezählers erfasst. Setzen Sie daher nach einem Austausch der Batterien den Batteriezähler zurück, um eine korrekte Erfassung der Gebrauchszeit zu gewährleisten.

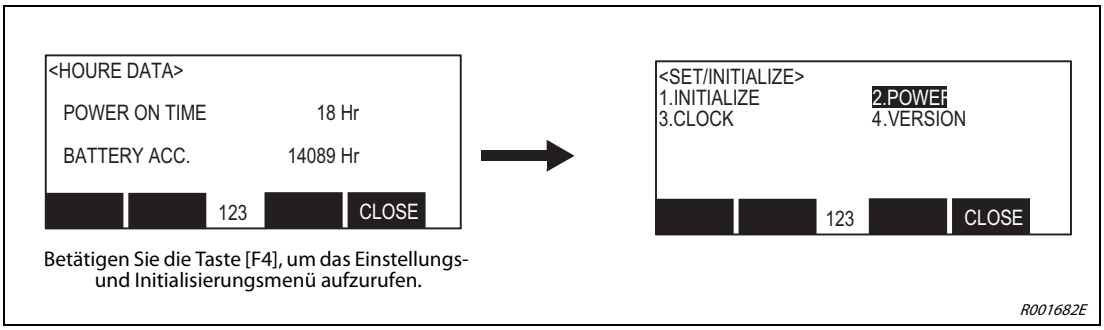
3.20.4 Verbleibende Betriebszeit der Batterie und Einschaltzeit anzeigen

Die Betriebszeit des Steuergerätes und die verbleibende Lebensdauer der Batterie in Stunden werden angezeigt.

① Betätigen Sie die Taste [2] im Einstellungs- und Initialisierungsmenü, um die verbleibende Betriebszeit der Batterie und Einschaltzeit anzuzeigen.



② Betätigen Sie die Funktionstaste, die der Funktion „CLOSE" zugeordnet ist, um zum Einstellungs- und Initialisierungsmenü zurückzukehren.

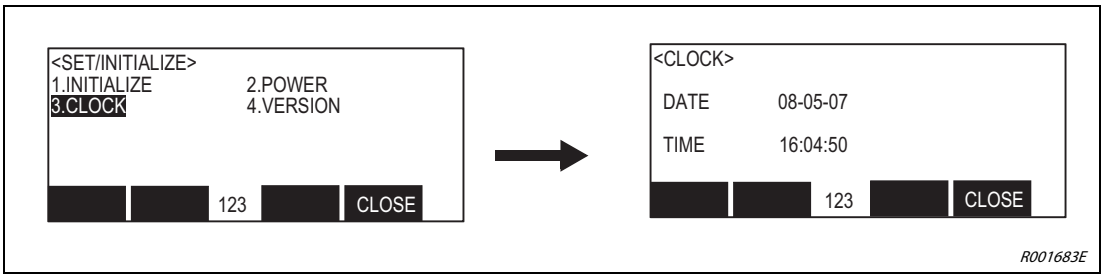


3.20.5 Datum und Uhrzeit einstellen

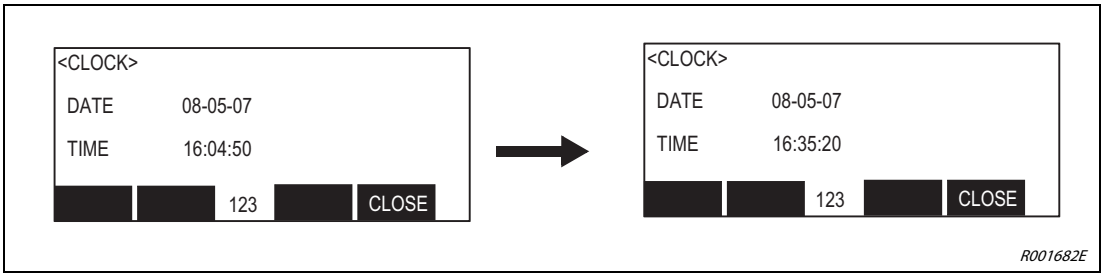
Das Steuergerät ist mit einer internen Uhr für Uhrzeit- und Datumsfunktionen ausgerüstet. Diese Datumsfunktion nutzt das Steuergerät z.B. zum Eintragen des Erstellungsdatums in ein Programm.

Die interne Uhr sollten Sie nach der erstmaligen Inbetriebnahme des Roboters und danach in regelmäßigen Abständen kontrollieren. Nachfolgend wird das Einstellen von Uhrzeit und Datum beschrieben.

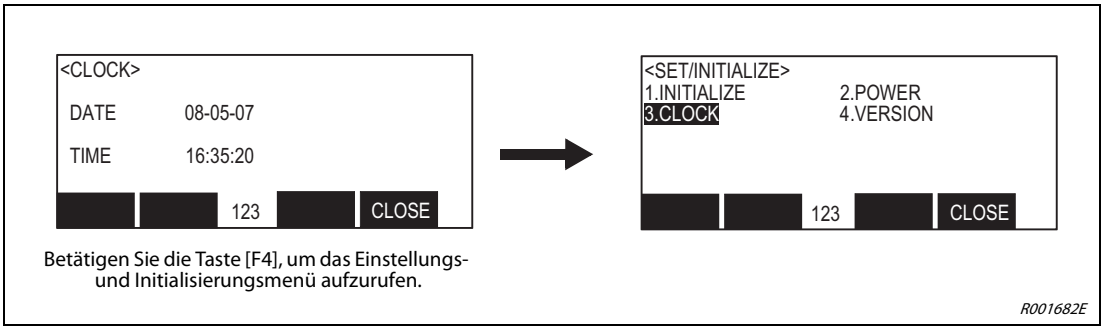
① Betätigen Sie die Taste [3] im Einstellungs- und Initialisierungsmenü, um das Menü zur Einstellung des Datums und der Uhrzeit aufzurufen.



② Bewegen Sie den Cursor mit Hilfe der Pfeiltasten zu dem Eintrag, den Sie ändern möchten, und stellen Sie das Datum und die Uhrzeit ein.

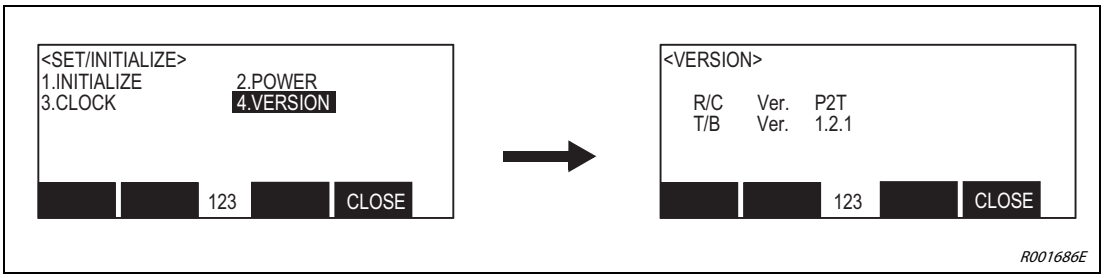


③ Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Einstellungs- und Initialisierungsmenü zurückzukehren.

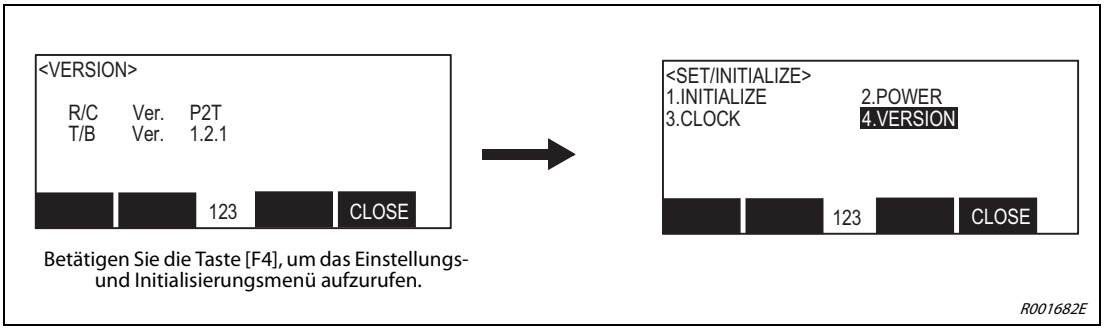


3.20.6 Software-Version der Teaching Box und des Steuergeräts anzeigen

① Betätigen Sie die Taste [4] im Einstellungs- und Initialisierungsmenü, um die Software-Version der Teaching Box und des Steuergeräts anzuzeigen.

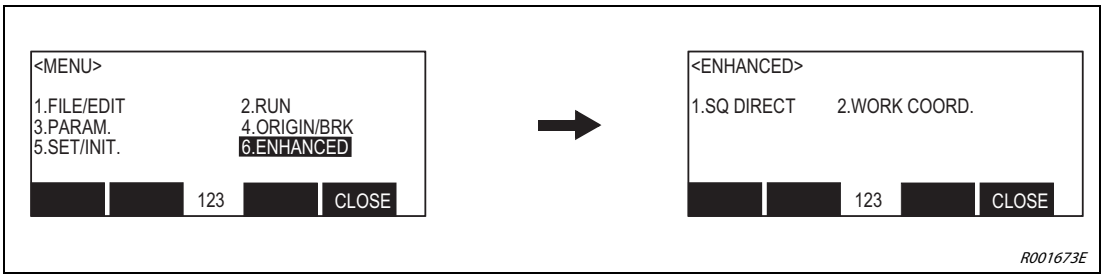


② Betätigen Sie die Funktionstaste, die der Funktion „CLOSE“ zugeordnet ist, um zum Einstellungs- und Initialisierungsmenü zurückzukehren.



3.21 Zusatzfunktionen

① Betätigen Sie die Taste [6] im Hauptmenü, um die Zusatzfunktionen aufzurufen.

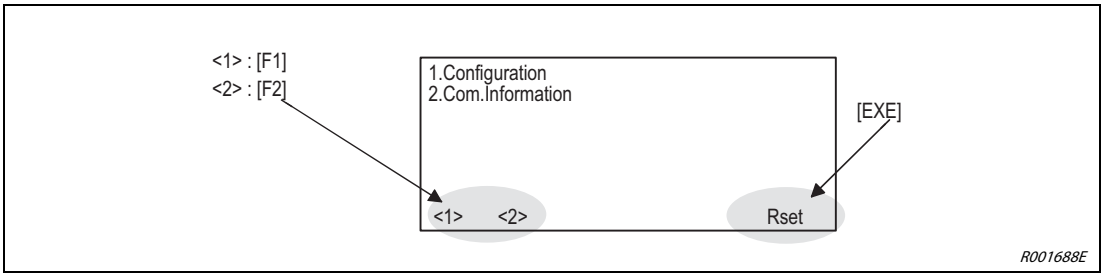


- SQ DIRECT
  Diese Funktion ermöglicht die direkte Steuerung eines Roboters der SQ-Serie über ein SPS-Programm.
- WORK COORD
  In diesem Menü können die Daten für ein Werkstück-Koordinatensystem vorgegeben werden. Definieren Sie hier Ihr Koordinatensystem für den Werkstück-JOG-Betrieb. Eine detaillierte Beschreibung des Werkstück-JOG-Betriebs finden Sie im Abschn. 3.4.8. Detaillierte Informationen zur Einstellung der Werkzeugkoordinaten finden Sie im Abschn. 9.2 und im Technischen Handbuch des Roboters.

3.22 Einstellungen im Startmenü

Im Startmenü kann die Sprache, in der die Anzeige auf der Teaching Box erfolgen soll, ausgewählt werden. Es stehen die Sprachen Japanisch und Englisch zur Verfügung.
Weiterhin ist der Kontrast der Anzeige in 16 Stufen einstellbar.

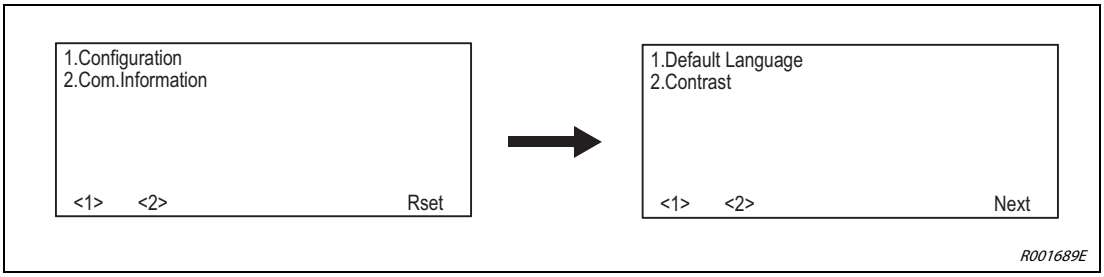
Nach dem Einschalten erscheint das Startmenü. Wählen Sie die entsprechende Funktion über die Tasten [F1] oder [F2].



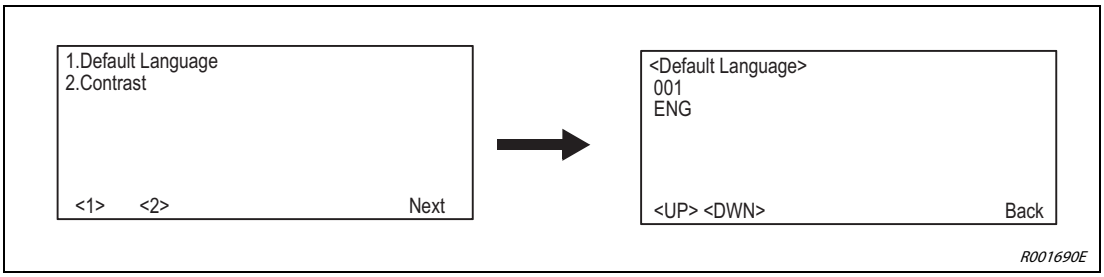
3.22.1 Auswahl der Landessprache

Stellen Sie die Landessprache, in der die Anzeige auf der Teaching Box erfolgen soll, ein. Es stehen die Sprachen Japanisch und Englisch zur Verfügung.

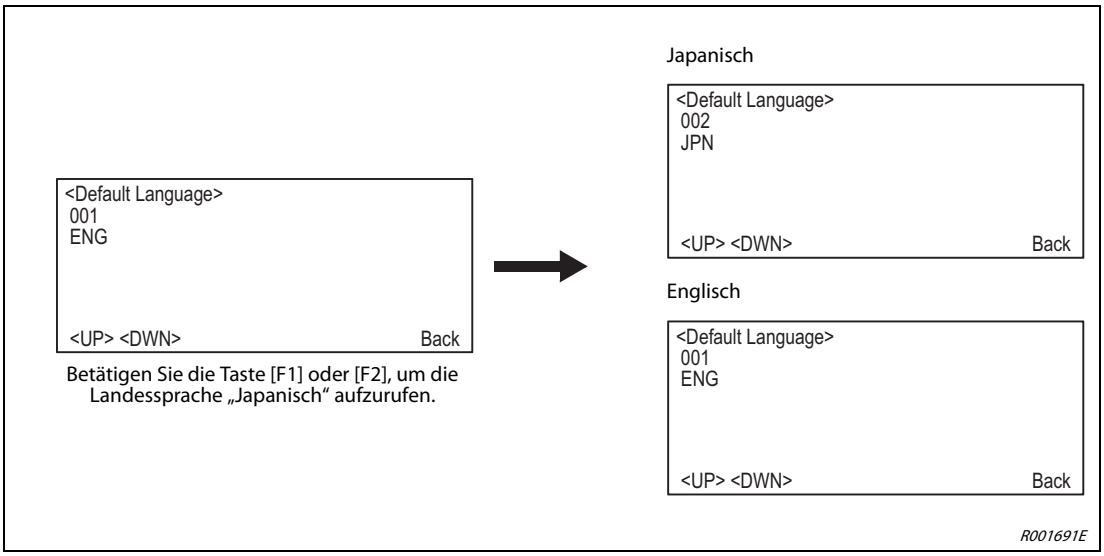
① Betätigen Sie die Taste [F1] im Startmenü, um das Konfigurationsmenü aufzurufen.



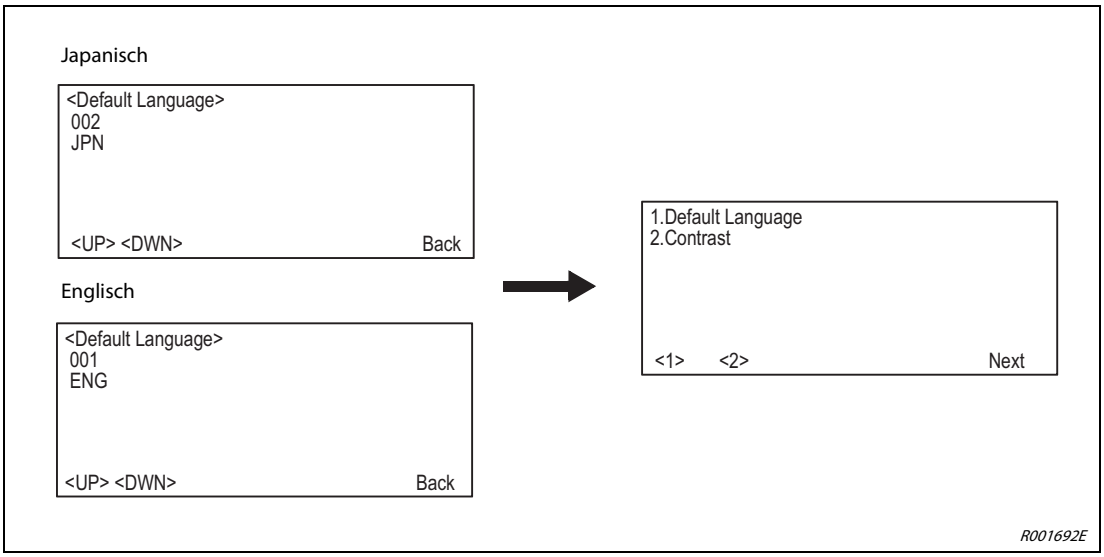
② Betätigen Sie die Taste [F1], um das Menü zur Einstellung der Landessprache aufzurufen.



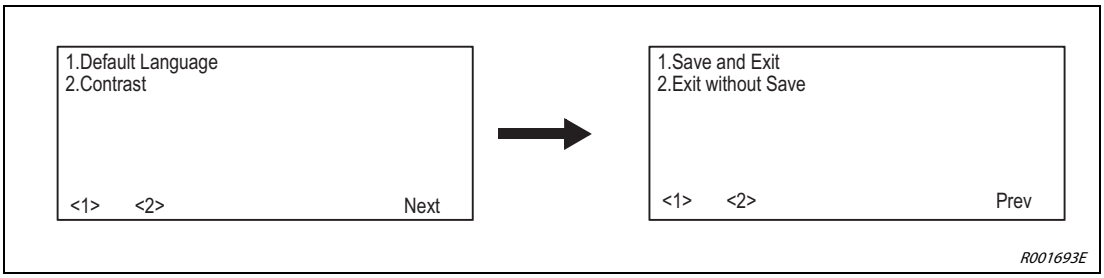
③ Betätigen Sie die Funktionstaste, die der Funktion „UP“ oder „DWN“ (down) zugeordnet ist, um die Landessprache zu wechseln.



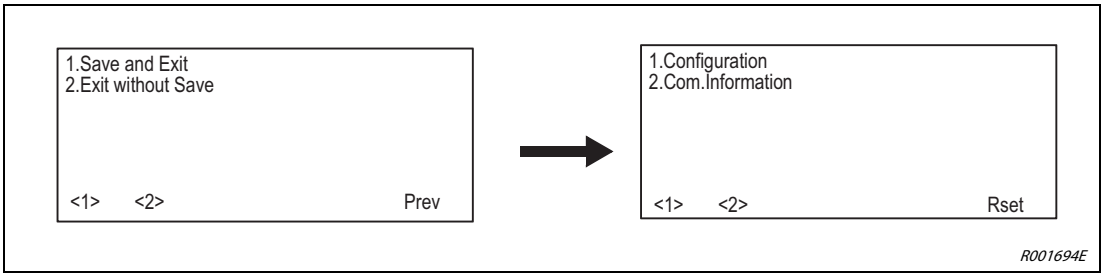
④ Betätigen Sie die Taste [EXE], um die Einstellung zu aktivieren.



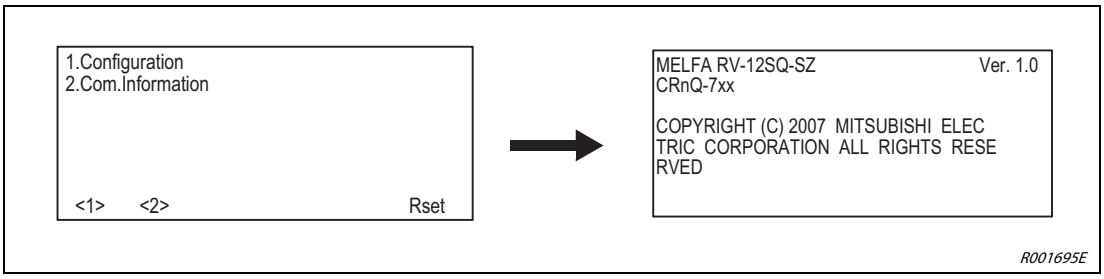
⑤ Betätigen Sie die Taste [EXE], um das Menü zu beenden.



⑥ Betätigen Sie die Taste [F1], um die Einstellung zu speichern. Möchten Sie die Einstellung nicht speichern, betätigen Sie die Taste [F2]. Das Startmenü erscheint.
Möchten Sie die Einstellung wiederholen, betätigen Sie die Taste [EXE].



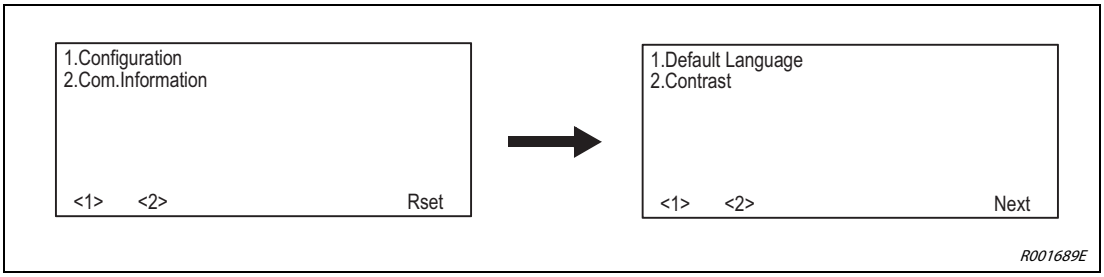
⑦ Die Anzeige der Teaching Box erscheint in der ausgewählten Landesprache.



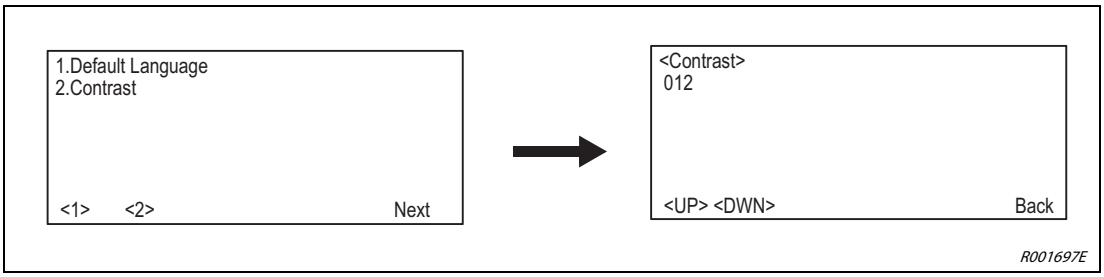
3.22.2 Einstellung des Kontrasts

Der Kontrast der Teaching-Box-Anzeige ist in 16 Stufen einstellbar. Je höher der eingestellte Wert, desto größer der Kontrast.

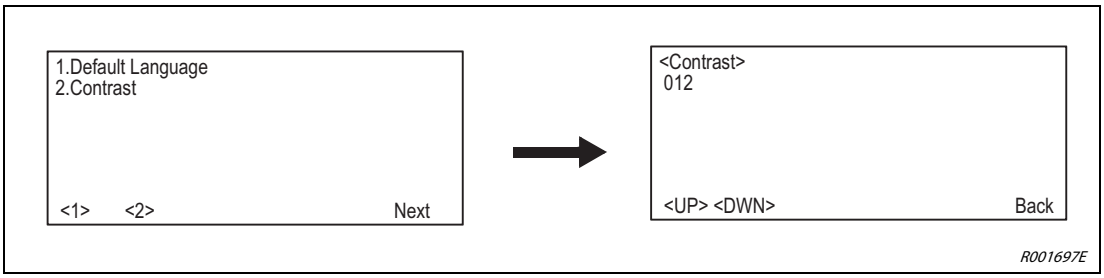
① Betätigen Sie die Taste [F1] im Startmenü, um das Konfigurationsmenü aufzurufen.



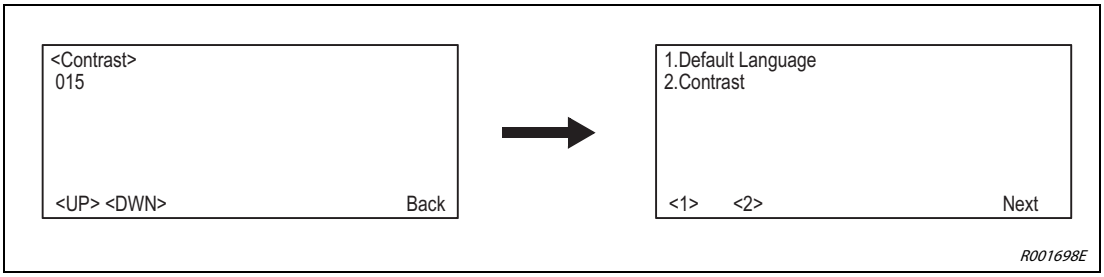
② Betätigen Sie die Taste [F2], um das Menü zur Einstellung des Kontrasts aufzurufen. Der Kontrast kann nun in einem Bereich von 0 bis 15 eingestellt werden.



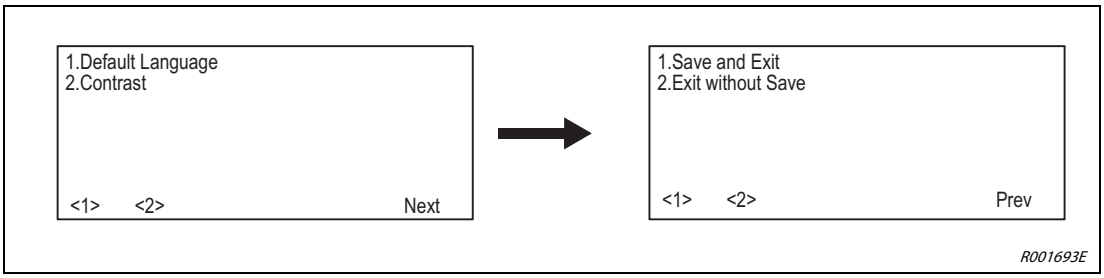
③ Betätigen Sie die Taste [F1], um den Kontrast zu vergrößern und die Taste [F2], um den Kontrast zu verkleinern.



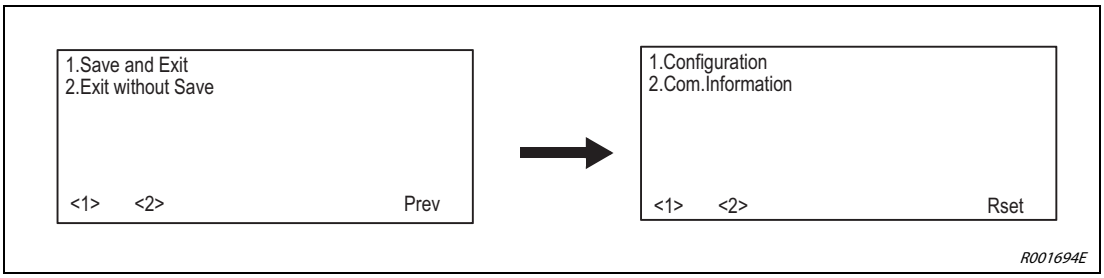
④ Betätigen Sie die Taste [EXE], um die Einstellung zu speichern.



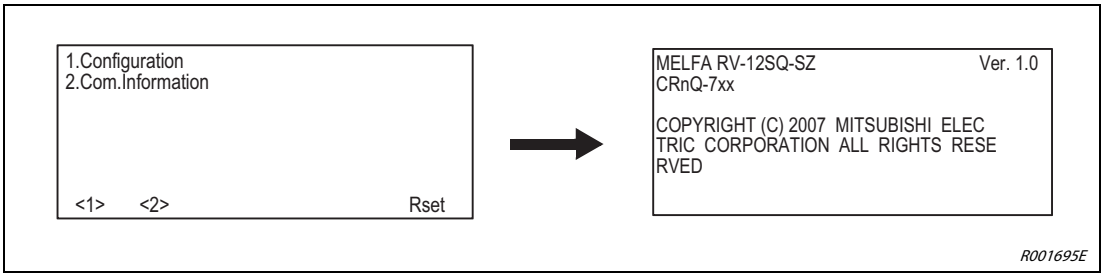
⑤ Betätigen Sie die Taste [EXE], um das Menü zu beenden.



⑥ Betätigen Sie die Taste [F1], um die Einstellung zu speichern. Möchten Sie die Einstellung nicht speichern, betätigen Sie die Taste [F2]. Das Startmenü erscheint.
Möchten Sie die Einstellung wiederholen, betätigen Sie die Taste [EXE].



⑦ Die Anzeige der Teaching Box erscheint mit dem ausgewählten Kontrast.



4 MELFA-BASIC-V-Programmierung

In diesem Kapitel finden Sie eine Einführung in die Programmiersprache MELFA-BASIC V. Eine detaillierte Beschreibung der einzelnen Befehle finden Sie in Kap. 6 „MELFA-BASIC-V-Befehle“.

4.1 Funktionsübersicht

Nr.	Zuordnung	Beschreibung	Befehl
1	Abschn. 4.3 „Steuerung der Roboterbewegung“	Abschn. 4.3.1 „Gelenk-Interpolation“	Mov
2		Abschn. 4.3.2 „Linear-Interpolation“	Mvs
3		Abschn. 4.3.3 „Kreis-Interpolation“	Mvr, Mvr2, Mvr3, Mvc
4		Abschn. 4.3.4 „Kontinuierliche Bewegung“	Cnt
5		Abschn. 4.3.5 „Beschleunigungs-/ Bremszeit und Geschwindigkeit“	Accel, Oadl
6		Abschn. 4.3.6 „Feinpositionierung“	Fine, Mov und Dly
7		Abschn. 4.3.7 „Verfahrweggenauigkeit“	Prec
8		Abschn. 4.3.8 „Hand- und Werkzeug-steuerung“	HOpen, HClose, Tool
9	Abschn. 4.4 „Palettierung“	—	Def Plt, Plt
10	Abschn. 4.5 „Programmsteuerung“	Abschn. 4.5.1 „Verzweigungen und Wartezeit“	GoTo, If Then Else, Wait usw.
11		Abschn. 4.5.2 „Programmschleife“	For Next, While WEnd
12		Abschn. 4.5.3 „Interrupt“	Def Act, Act
13		Abschn. 4.5.4 „Unterprogramm“	GoSub, CallP, On GoSub usw.
14		Abschn. 4.5.5 „Timer“	Dly
15		Abschn. 4.5.6 „Stopp“	End (1Zyklus Pause), Hlt
16	Abschn. 4.6 „Ein- und Ausgabe externer Signale“	Abschn. 4.6.1 „Eingangssignale“	M_In, M_Inb, M_Inw usw.
17		Abschn. 4.6.2 „Ausgangssignale“	M_Out, M_Outb, M_Outw usw.
18	Abschn. 4.7 „Kommunikation“	—	Open, Close, Print, Input usw.
19	Abschn. 4.8 „Ausdrücke und Operationen“	Abschn. 4.8.1 „Übersicht der Operationen“	+, –, *, /, <>, <, > usw.
20		Abschn. 4.8.2 „Relative Konvertierung (Multiplikation)“	P1 * P2
21		Abschn. 4.8.2 „Relative Konvertierung (Addition)“	P1 + P2
22	Abschn. 4.9 „Angehängte Anweisung“	—	Wth, WthIf

Tab. 4-1: *Übersicht der MELFA-BASIC-V-Funktionen*

4.2 Programmaufbau

In diesem Abschnitt werden die wichtigsten Elemente zum Aufbau eines Programms erläutert. Eine detaillierte Erklärung der einzelnen Begriffe finden Sie im Kap. 5.

4.2.1 Programmname

Ein Programmname darf aus maximal 12 Zeichen bestehen. Auf der Anzeige des Steuergeräts können jedoch nur bis zu 4 Zeichen dargestellt werden. Es empfiehlt sich daher, bei der Vergabe von Programmnamen nur 4 Zeichen zu verwenden.

Verwendbare Zeichen für Programmnamen	
Buchstaben	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ①
Zahlen	0 1 2 3 4 5 6 7 8 9

Tab. 4-2: *Für Programmnamen zugelassene Zeichen*

① Verwenden Sie in Programmnamen nur Großbuchstaben. Die Verwendung von Kleinbuchstaben kann zu einer fehlerhaften Abarbeitung des Programmes führen.

HINWEIS

Es ist nicht möglich ein Programm mit einem Programmnamen, der aus mehr als 4 Zeichen besteht, über das Steuergerät auszuwählen. Soll ein auszuführendes Programm über ein externes Ausgangssignal ausgewählt werden, sind im Programmnamen nur Zahlen zu verwenden. Bei einem Programm, dass über den Befehl CALLP ausgeführt werden soll, darf der Programmname aus mehr als 4 Zeichen bestehen.

4.2.2 Anweisung

In diesem Abschnitt werden die Elemente zum Aufbau einer Anweisung erläutert.

```
1    Mov P1    Wth M_Out(17) = 1
❶    ❷  ❸      ❹
```

❶ Schrittnummer
Für die einwandfreie Funktion eines Programms müssen die Schrittnummern in aufsteigender Reihenfolge angeordnet sein. Das Programm wird in dieser Reihenfolge abgearbeitet.

❷ Befehl
Der Befehl legt die Aktion des Roboters fest.

❸ Befehlsparameter
Der Befehlsparameter kann z.B. eine Variable oder ein Wert sein.

❹ Angehängte Anweisung
Bei Interpolationsbefehlen ist es möglich, eine Verknüpfung an die Anweisung anzuhängen. Durch Anhängen einer Verknüpfung können bestimmte Befehle parallel zum Interpolationsbefehl ausgeführt werden.

4.2.3 Variable

Folgende Variablen können in einem Programm verwendet werden:

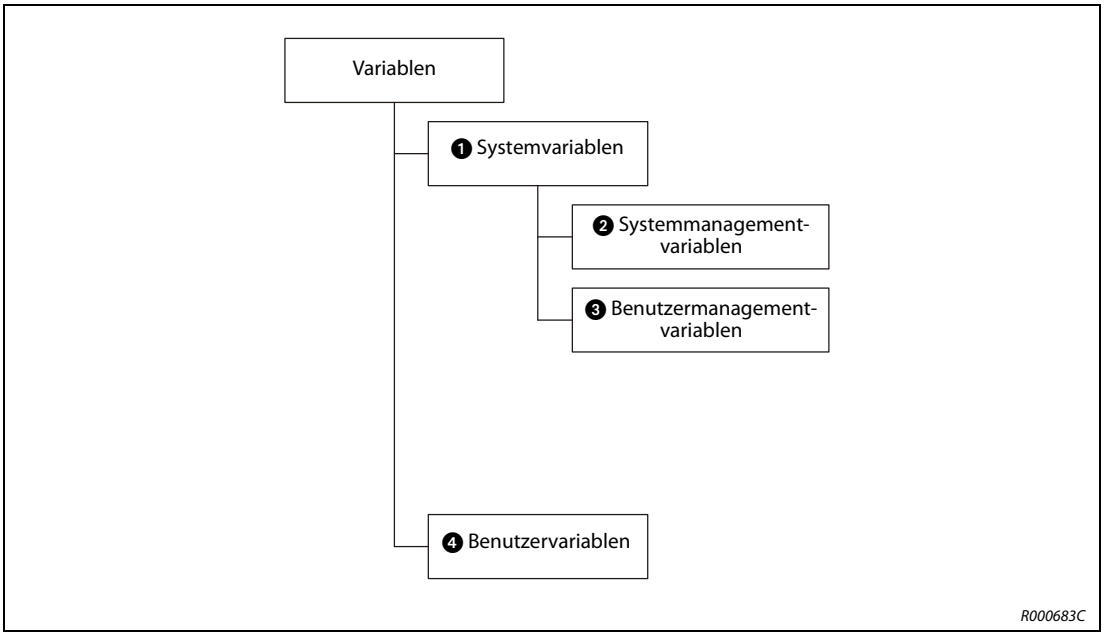


Abb. 4-1: *Einteilung der Variablen*

❶ Systemvariablen sind durch einen Variablennamen und einen gespeicherten Wert definiert.

❷ Systemmanagementvariablen können nur gelesen werden.
Beispiel: P_CURR
In dieser Variablen wird die aktuelle Position des Roboters ständig gespeichert.

❸ Benutzermanagementvariablen können gelesen und geschrieben werden. Eingangssignale können nur gelesen werden.
Beispiel: M_OUT(17) = 1: Ausgangsbit 17 einschalten
M1 = M_IN(20): Schreibe den Wert des Eingangsbits 20 in die numerische Variable M1.

❹ Benutzervariablen sind durch einen Variablennamen und den Verwendungszweck definiert.

Jede Variable der oben aufgeführten Variablentypen ist einer der folgenden Gruppen zugeordnet:

- Positionsvariablen
  Eine Positionsvariable enthält die kartesischen Koordinaten des Roboters. Der Variablennamen beginnt mit „P“.
  Beispiel: Mov P1 Der Roboter fährt die Position an, die in der Variablen P1 abgespeichert ist.

- Gelenkvariablen
  Eine Gelenkvariable enthält die Winkelwerte der Robotergelenke. Der Variablenname beginnt mit „J“.
  Beispiel: Mov J1 Der Roboter fährt die Position an, die in der Variablen J1 abgespeichert ist.

- Numerische Variablen
  Eine numerische Variable enthält einen numerischen Wert (Integer, Reelle Zahl, usw.). Der Variablennamen beginnt mit „M“.
  Beispiel: M1 = 1 Der Wert „1“ wird in die Variable M1 geschrieben.

- Zeichenkettenvariablen
  Eine Zeichenkettenvariable enthält eine Zeichenkette. Dem Variablennamen folgt das Zeichen „$“.
  Beispiel: C1$ = "ERROR" Die Zeichenkette „ERROR“ wird in die Variable C1$ geschrieben.

4.3 Steuerung der Roboterbewegung

4.3.1 Gelenk-Interpolation

Die Handspitze wird mittels Gelenk-Interpolation zu einer festgelegten Position bewegt.

Erläuterung

Befehl	Beschreibung
Mov	Bewegt die Handspitze mittels Gelenk-Interpolation zu einer festgelegten Position Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen Wth oder WthIf erlauben das Anhängen einer Anweisung.

Anweisungsbeispiele

Mov P1	Position P1 anfahren
Mov P1 + P2	Position anfahren, die sich aus der Addition der Koordinaten der Positionen P1 und P2 ergibt
Mov P1 * P2	Position anfahren, die sich aus der relativen Konvertierung von P1 zu P2 ergibt
Mov P1, –50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P1 entfernt ist **(siehe Achtungshinweis)**
Mov P1 Wth M_Out(17) = 1	Position P1 anfahren und Ausgangsbit 17 auf „1“ setzen
Mov P1 WthIf M_In(20) = 1, Skip	Wird beim Anfahren der Position P1 das Eingangsbit 20 auf „1“ gesetzt, wird die Verfahrbewegung unterbrochen und das Programm wird in der nächsten Zeile fortgesetzt.
Mov P1 Type 1, 0	Position P1 indirekt (oder direkt) anfahren, wenn die Achsendrehungen größer als 180° sind (Grundeinstellung: indirekte Anfahrt)

Direkte und indirekte Anfahrt

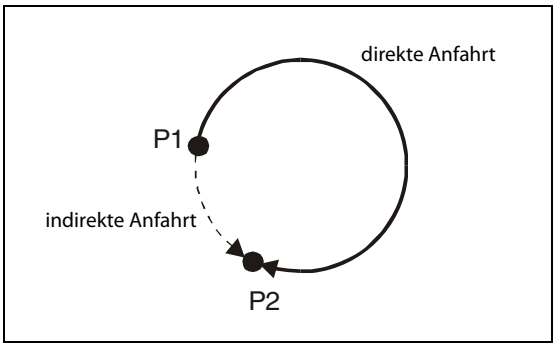


Abb. 4-2:
Direkte und indirekte Anfahrt einer Position

R000916C

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Programmbeispiel

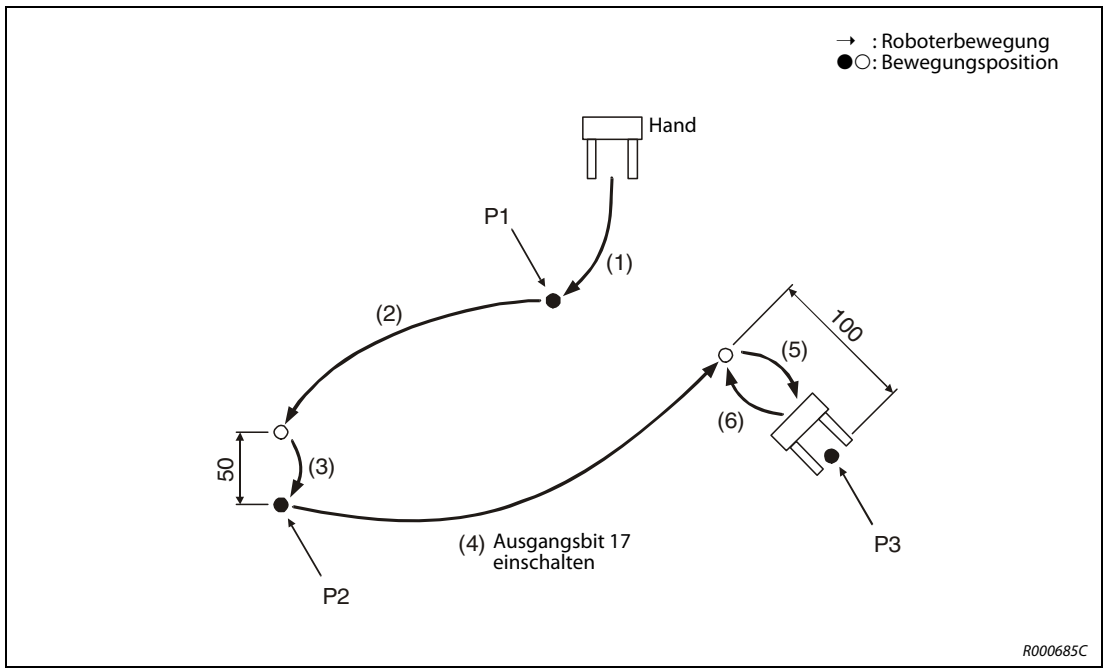


Abb. 4-3: *Verlauf des Verfahrweges bei Gelenk-Interpolation*

```
1  Mov P1                          'Position P1 anfahren
2  Mov P2, −50                     'Position anfahren, die 50 mm in Werkzeug-
                                   längsrichtung von der Position P2 entfernt ist
                                   (siehe Achtungshinweis)
3  Mov P2                          'Position P2 anfahren
4  Mov P3, −100 Wth M_Out(17) = 1  'Position anfahren, die 100 mm in Werkzeug-
                                   längsrichtung von Position P3 entfernt
                                   ist und Ausgangsbit 17 auf „1“ setzen
5  Mov P3                          'Position P3 anfahren
6  Mov P3, −100                    'Position anfahren, die 100 mm in Werkzeug-
                                   längsrichtung von Position P3 entfernt ist
7  End                             'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Angehängte Anweisung	⇒	Abschn. 6.3.82

4.3.2 Linear-Interpolation

Die Handspitze wird mittels Linear-Interpolation zu einer festgelegten Position bewegt.

Erläuterung

Befehl	Beschreibung
Mvs	Bewegt die Handspitze mittels Linear-Interpolation zu einer festgelegten Position Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen WTH oder WTHIF erlauben das Anhängen einer Anweisung.

Anweisungsbeispiele

Mvs P1	Position P1 anfahren
Mvs P1 + P2	Position anfahren, die sich aus der Addition der Koordinaten der Positionen P1 und P2 ergibt
Mvs P1 * P2	Position anfahren, die sich aus der relativen Konvertierung von P1 zu P2 ergibt
Mvs P1, −50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P1 entfernt ist **(siehe Achtungshinweis)**
Mvs P1 Wth M_Out(17) = 1	Position P1 anfahren und Ausgangsbit 17 auf „1“ setzen
Mvs P1 Wthlf M_In(20) = 1, Skip	Wird beim Anfahren der Position P1 das Eingangsbit 20 auf „1“ gesetzt, wird die Verfahrbewegung unterbrochen und das Programm wird in der nächsten Zeile fortgesetzt.
Mvs P1 Type 0, 0	Position P1 mittels Drehung anfahren
Mvs P1, TYPE 0, 1	Position P1 mittels orthogonaler 3-Achsen-Interpolation anfahren

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Programmbeispiel

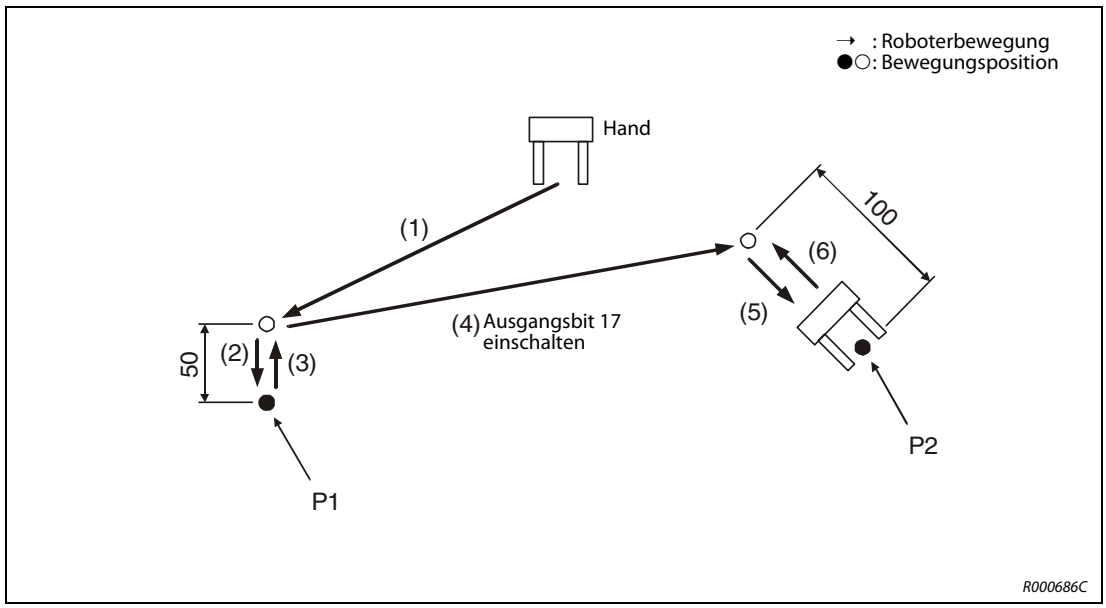


Abb. 4-4: *Verlauf des Verfahrweges bei Linear-Interpolation*

```
1  Mvs P1, –50                    'Position mittels Linear-Interpolation
                                  anfahren, die 50 mm in Werkzeuglängsrichtung
                                  von der Position P1 entfernt ist
                                  (siehe Achtungshinweis)
2  Mvs P1                         'Position P1 mittels Linear-Interpolation anfahren
3  Mvs, –50                       'Position mittels Linear-Interpolation
                                  anfahren, die 50 mm in Werkzeuglängsrichtung
                                  von der aktuellen Position
                                  (P1) entfernt ist (siehe Achtungshinweis)
4  Mvs P2, –100 Wth M_Out(17) = 1 'Position mittels Linear-Interpolation
                                  anfahren, die 100 mm in Werkzeug-
                                  längsrichtung von Position P2 entfernt
                                  ist und Ausgangsbit 17 auf „1“ setzen
5  MVS P2                         'Position 2 mittels Linear-Interpolation anfahren
6  Mvs, –100                      'Position mittels Linear-Interpolation
                                  anfahren, die 100 mm in Werkzeuglängsrichtung
                                  von der aktuellen Position
                                  (P2) entfernt ist (siehe Achtungshinweis)
7  End                            'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4
Gelenk-Interpolation	⇒	Abschn. 4.3.1
Kreis-Interpolation	⇒	Abschn. 4.3.3
Angehängte Anweisung	⇒	Abschn. 6.3.82

4.3.3 Kreis-Interpolation

Die Handspitze wird mittels 3D-Kreis-Interpolation entlang eines durch 3 Punkte festgelegten Kreises zu einer festgelegten Position bewegt. Entspricht die aktuelle Position nicht der Startposition, wird die Startposition mittels Linear-Interpolation angefahren.

Erläuterung

Befehl	Beschreibung
Mvr	Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch die Startposition, Zwischenposition und Endposition festgelegten Kreisbogens Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen Wth oder WthIf erlauben das Anhängen einer Anweisung.
Mvr2	Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition Der Kreisbogen wird durch die Startposition, die Referenzposition und die Endposition festgelegt. Die Roboterbewegung geht dabei nicht durch den Referenzpunkt. Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen Wth oder WthIf erlauben das Anhängen einer Anweisung.
Mvr3	Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition Der Kreisbogen wird durch die Startposition, den Mittelpunkt und die Endposition festgelegt. Der Zentriwinkel zwischen Start- und Endposition liegt dabei zwischen 0° und 180°. Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen Wth oder WthIf erlauben das Anhängen einer Anweisung.
Mvc	Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch Startposition (Endposition), Zwischenposition 1, Zwischenposition 2 und Endposition festgelegten Kreisbogens Mit Hilfe der Verknüpfungen WTH oder WTHIF kann eine Anweisung angehängt werden.

Anweisungsbeispiele

Mvr P1, P2, P3	Bewegung entlang des Kreisbogens P1 → P2 → P3
Mvr P1, P2, P3 Wth M_Out(17) = 1	Bewegung entlang des Kreisbogens P1 → P2 → P3 und Ausgangsbit 17 auf „1“ setzen
Mvr P1, P2, P3 WthIf M_In(20) = 1, SKIP	Fährt entlang des Kreisbogens P1 → P2 → P3 und unterbricht die Bewegung, wenn Eingangsbit 20 auf „1“ gesetzt wird Die Programmsteuerung springt in die nächste Zeile.
Mvr P1, P2, P3 TYPE 0, 1	Bewegung entlang des Kreisbogens P1 → P2 → P3
Mvr2 P1, P3, P11	Bewegung entlang des Kreisbogens von P1 nach P3, ohne die Referenzposition P11 zu durchlaufen
Mvr3 P1, P3, P10	Bewegung entlang des Kreisbogens von P1 nach P3 in Richtung des kleineren Zentriwinkels P10 ist der Mittelpunkt.
Mvc P1, P2, P3	Bewegung entlang des Kreisbogens P1 → P2 → P3

Programmbeispiel

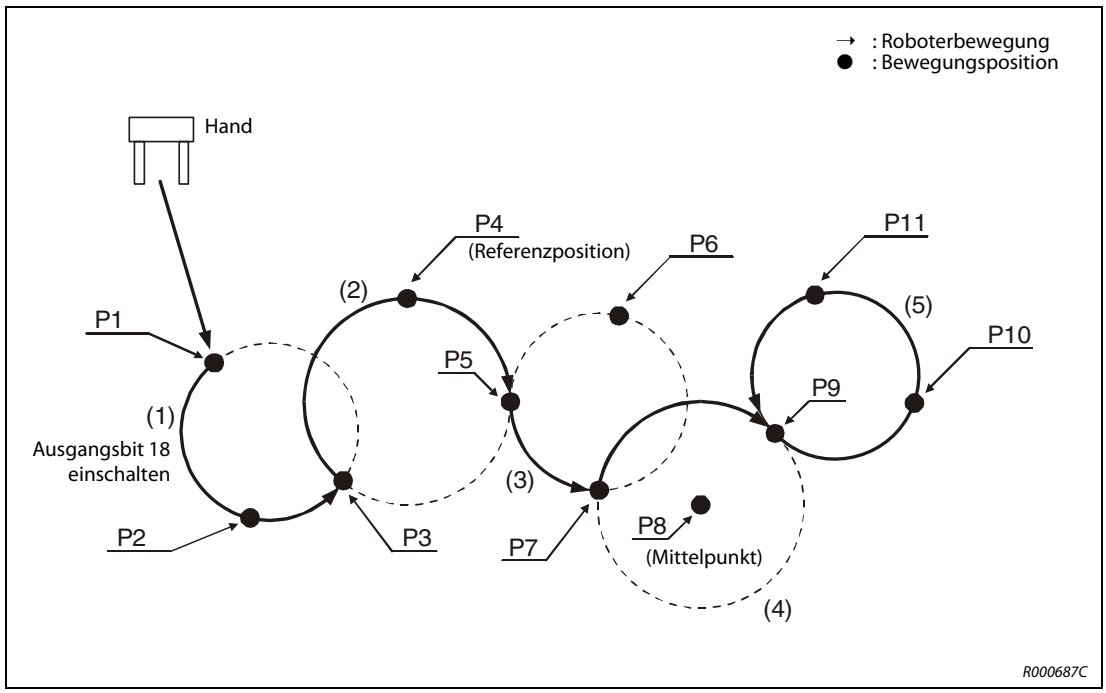


Abb. 4-5: *Verlauf des Verfahrweges bei Kreis-Interpolation*

```
1  Mvr P1, P2, P3 Wth M_Out(18) = 1   'Bewegung entlang des Kreisbogens
                                       P1 → P2 → P3
                                       Die aktuelle Position entspricht nicht der
                                       Startposition. Der Roboter bewegt sich also
                                       zuerst mittels Linear-Interpolation zur
                                       Startposition (P1).
                                       Mit Beginn der Kreis-Interpolation wird das
                                       Ausgangsbit 18 auf „1“ gesetzt.
2  Mvr P3, P4, P5                      'Bewegung entlang des Kreisbogens
                                       P3 → P4 → P5
3  Mvr2 P5, P7, P6                     'Bewegung entlang des Kreisbogens
                                       von P5 nach P7, ohne die Referenzposition
                                       P6 zu durchlaufen
4  Mvr3 P7, P9, P8                     'Bewegung entlang des Kreisbogens von P7 nach P9
                                       in Richtung des kleineren Zentriwinkels
                                       P8 ist der Mittelpunkt.
5  Mvc P9, P10, P11                    'Bewegung entlang des Kreisbogens
                                       P9 → P10 → P11 → P9
                                       Entspricht die aktuelle Position nicht der
                                       Startposition, wird die Startposition mittels
                                       Linear-Interpolation angefahren (1 Zyklus).
6  End                                 'Programmende
```

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4
Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Angehängte Anweisung	⇒	Abschn. 6.3.82

4.3.4 Kontinuierliche Bewegung

Bei freigegebener Cnt-Einstellung fährt der Roboter die festgelegten Positionen ohne zu stoppen an. Der Cnt-Befehl definiert den Start- bzw. Endpunkt der kontinuierlichen Bewegung. Die Geschwindigkeit kann während der kontinuierlichen Bewegung verändert werden.

Erläuterung

Befehl	Beschreibung
Cnt	Legt den Start- und Endpunkt für die kontinuierliche Bewegung fest

Anweisungsbeispiele

Cnt 1	Freigeben der Cnt-Einstellung
Cnt 1, 100, 200	Freigeben der Cnt-Einstellung Der Anfangspunktabstand der kontinuierlichen Bewegung beträgt 100 mm und der Endpunktabstand der kontinuierlichen Bewegung beträgt 200 mm.
Cnt 0	Sperren der Cnt-Einstellung

Programmbeispiel

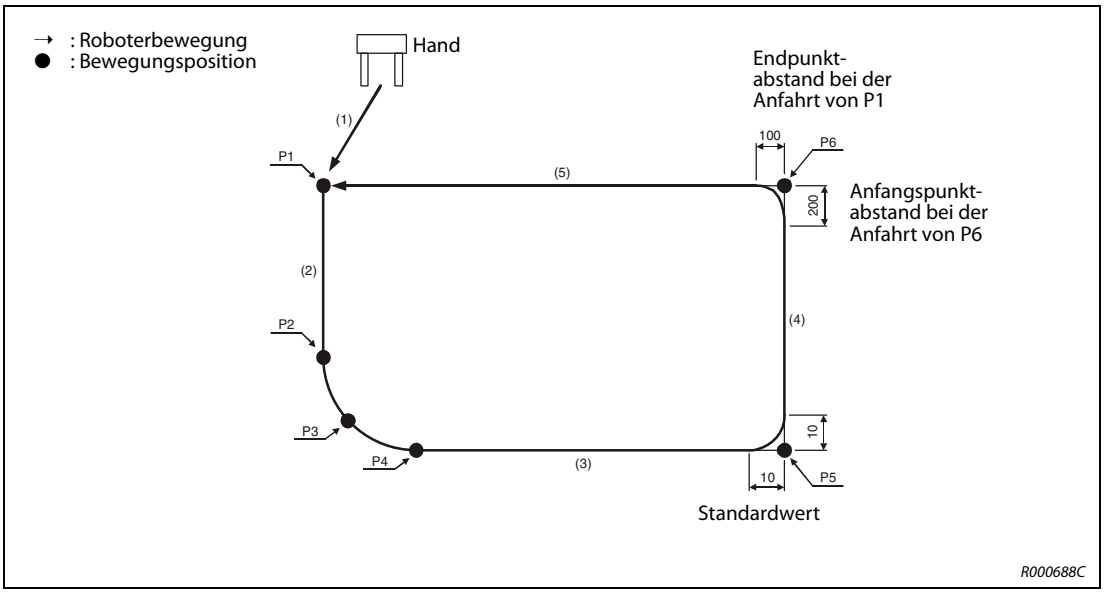


Abb. 4-6: *Verlauf des Verfahrweges bei kontinuierlicher Bewegung*

```
1 Mov P1          'Position P1 mittels Gelenk-Interpolation anfahren
2 Cnt 1           'Cnt-Einstellung freigeben
                  Cnt1 (10,10) ⇒ Standardwert 10, 10
                  (Alle folgenden Bewegungen sind kontinuierlich.)
3 Mrv P2, P3, P4  'Position P2 mittels Linear-Interpolation und Position P4
                  kontinuierlich mittels Kreis-Interpolation anfahren
4 Mvs P5          'Position P5 mittels Linear-Interpolation anfahren
5 Cnt 1, 200, 100 'Anfangspunktabstand der kontinuierlichen Bewegung
                  auf 200 mm und Endpunktabstand der kontinuierlichen
                  Bewegung auf 100 mm festlegen
6 Mvs P6          'Nach Erreichen von Position P5, Position P6 mittels
                  Linear-Interpolation anfahren
7 Mvs P1          'Position P1 kontinuierlich mittels Linear-Interpolation
                  anfahren
8 Cnt 0           'Cnt-Einstellung sperren
9 End             'Programmende
```

ACHTUNG:

Der genaue Verfahrweg des Roboters kann sich in Abhängigkeit der Geschwindigkeit ändern. Besonders im Bereich der Eckpunkte kann der Verfahrweg variieren. Verfahren Sie den Roboter beim Start des Automatikbetriebs daher erst mit niedriger Geschwindigkeit. Steigern Sie die Geschwindigkeit allmählich und achten Sie darauf, dass keine Kollisionen mit umliegenden Einheiten auftreten.

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3

4.3.5 Beschleunigungs-/Bremszeit und Geschwindigkeit

Die Beschleunigung/Abbremsung kann bezogen auf den Maximalwert eingestellt werden. Die Geschwindigkeit ist ebenfalls einstellbar.

Erläuterung

Befehl	Beschreibung
Accel	Die Beschleunigung/Abbremsung kann bezogen auf den Maximalwert (%) eingestellt werden.
Ovrd	Die Geschwindigkeit für das gesamte Programm kann bezogen auf den Maximalwert (%) eingestellt werden.
JOvrd	Die Geschwindigkeit für die Gelenk-Interpolation kann bezogen auf den Maximalwert (%) eingestellt werden.
Spd	Legt die Geschwindigkeit (mm/s) für Linear- und Kreis-Interpolation fest
Oadl	Freigabe der Einstellung für die optimalen Beschleunigung/Abbremsung

Anweisungsbeispiele

Accel	Setzt die Beschleunigung und die Abbremsung auf 100 %
Accel 60, 80	Setzt die Beschleunigung auf 60 % und die Abbremsung auf 80 % des Maximalwertes (Bei einer maximalen Beschleunigungs-/Bremszeit von 0,2 s ergibt sich eine Beschleunigungszeit von 0,33 s und eine Bremszeit von 0,25 s.)
Ovrd 50	Legt die Geschwindigkeit für Gelenk-, Linear- und Kreis-Interpolation auf 50 % der maximalen Geschwindigkeit fest
JOvrd 70	Legt die Geschwindigkeit für Gelenk-Interpolation auf 70 % der maximalen Geschwindigkeit fest
Spd 30	Legt die Geschwindigkeit für Linear- und Kreis-Interpolation auf 30 mm/s fest
Oadl ON	Gibt die optimale Beschleunigung/Abbremsung frei

Die aktuelle Arbeitsgeschwindigkeit ergibt sich:

Gelenk-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des Ovrd-Befehls	×	Einstellwert des JOvrd-Befehls
Linear-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des Ovrd-Befehls	×	Einstellwert des Spd-Befehls

Programmbeispiel

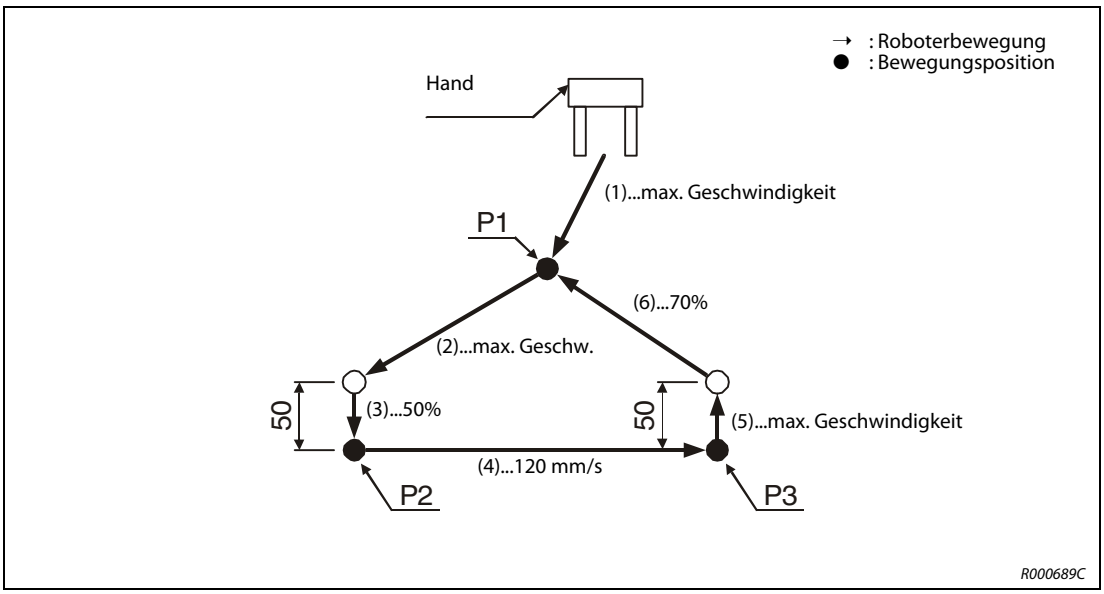


Abb. 4-7: *Verfahrweg und Geschwindigkeiten*

```
1  Ovrd 100          'Legt die Geschwindigkeit für das gesamte Programm auf den
                     Maximalwert fest
2  Mvs P1            '(1) Position P1 mit Maximalgeschwindigkeit anfahren
3  Mvs P2, –50       '(2) Position mit Maximalgeschwindigkeit anfahren, die 50 mm in
                     Werkzeuglängsrichtung von der Position P2 entfernt ist
                     (siehe Achtungshinweis)
4  Ovrd 50           'Legt die Geschwindigkeit für das gesamte Programm auf den
                     halben Maximalwert fest
5  Mvs P2            '(3) Position P2 mittels Linear-Interpolation und halber Maximal-
                     geschwindigkeit anfahren
6  Spd 120           'Legt die Endgeschwindigkeit auf 120 mm/s fest
                     (Mit dem OVRD-Wert von 50 % ergibt sich eine aktuelle
                     Geschwindigkeit von 60 mm/s.)
7  Ovrd 100'         'Legt die Geschwindigkeit auf 100 % fest, so dass eine End-
                     geschwindigkeit von 120 mm/s erreicht wird
8  Accel 70, 70      'Die Beschleunigung/Abbremsung wird auf 70 % des Maximalwerts
                     gesetzt
9  Mvs P3            '(4) Position P3 mittels Linear-Interpolation und mit einer
                     Endgeschwindigkeit von 120 mm/s anfahren
10 Spd M_NSpd        'Setzt die Geschwindigkeit auf den Standardwert zurück
11 JOvrd 70          'Legt die Geschwindigkeit für Gelenk-Interpolation auf 70 % fest
12 Accel             'Legt die Beschleunigung/Abbremsung auf 100 % fest
13 Mvs, –50          '(5) Position mittels Linear-Interpolation und Standard-
                     geschwindigkeit anfahren, die 50 mm in Werkzeuglängsrichtung
                     von der aktuellen Position (P3) entfernt ist (siehe Achtungshinweis)
14 Mov P1            '(6) Position P1 mit 70 % der Maximalgeschwindigkeit anfahren
15 End               'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4

4.3.6 Feinpositionierung

Der Abschluss eines Positioniervorgangs wird durch eine Anzahl von Impulsen festgelegt. Die Einstellung ist bei Ausführung kontinuierlicher Bewegungen deaktiviert.

Erläuterung

Befehl	Beschreibung
Fine	Legt den Abschluss eines Positioniervorgangs durch eine Anzahl von Impulsen fest
Mov und Dly	Nach einem Mov-Befehl kann der Abschluss des Positioniervorgangs auch durch einen Dly-Befehl (Timer) erfolgen.

Anweisungsbeispiele

Fine 100	Legt die Anzahl der Impulse zur Feinpositionierung auf 100 fest
Mov P1	Position P1 anfahren (Die Verfahrbewegung ist bei dem Befehlswert abgeschlossen.)
Dly 0.1	Der Abschluss des Positioniervorgangs erfolgt über die Timer-Einstellung.

Programmbeispiel

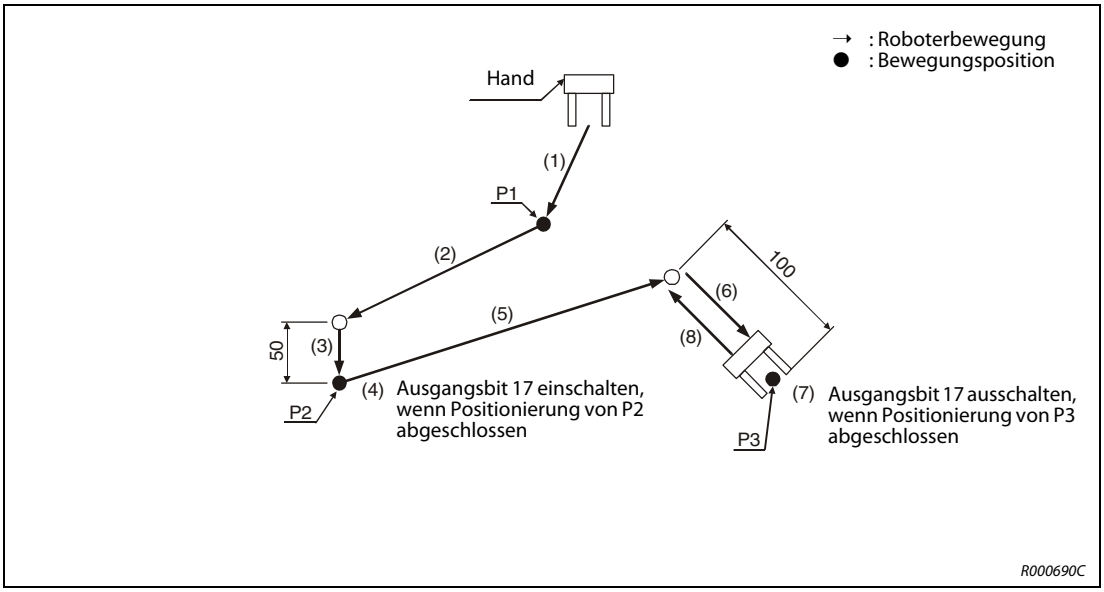


Abb. 4-8: *Verfahrweg und Feinpositionierung*

```
1  Cnt 0              'Die Feinpositionierung ist nur freigegeben, wenn die Einstellung
                      für kontinuierliche Bewegungen deaktiviert ist.
2  Mvs P1             '(1) Position P1 mittels Linear-Interpolation anfahren
3  Mvs P2, -50        '(2) Position mit Linear-Interpolation anfahren, die 50 mm in
                      Werkzeuglängsrichtung von der Position P2 entfernt ist
                      (siehe Achtungshinweis)
4  Fine 50            'Legt die Anzahl der Impulse zur Feinpositionierung auf 50 fest
5  Mvs P2             '(3) Position P2 mittels Linear-Interpolation anfahren
                      (Die Positionierung ist bei einer Impulszahl von kleiner gleich 50
                      abgeschlossen.)
6  M_OUT(17) = 1      '(4) Ausgangsbit 17 wird auf „1“ gesetzt, wenn die Anzahl der
                      Impulse 50 erreicht.
7  Fine 1000          'Legt die Anzahl der Impulse zur Feinpositionierung auf 1000 fest
8  Mvs P3, -100       '(5) Position mit Linear-Interpolation anfahren, die 100 mm in
                      Werkzeuglängsrichtung von der Position P3 entfernt ist
                      (siehe Achtungshinweis)
9  Mvs P3             '(6) Position P3 mittels Linear-Interpolation anfahren
10 Dly 0.1            'Die Positionierung erfolgt über Timer.
11 M_OUT(17) = 0      (7) Ausgangsbit 17 wird auf „0“ gesetzt.
12 Accel              'Legt die Beschleunigung/Abbremsung auf 100 % fest
13 Mvs, -100          '(8) Position mit Linear-Interpolation anfahren, die 100 mm in
                      Werkzeuglängsrichtung von der aktuellen Position (P3) entfernt ist
                      (siehe Achtungshinweis)
14 End                'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4

4.3.7 Verfahrweggenauigkeit

Die Verfahrwegtreue bei der Ausführung von Bewegungsbefehlen kann erhöht werden. Die Funktion ist nur für bestimmte Robotermodelle verfügbar. Zur Zeit kann der Befehl mit 5- oder 6-achsigen Vertikal-Knickarmrobotern verwendet werden.

Erläuterung

Befehl	Beschreibung
Prec	Legt die Präzision des Verfahrwegs fest

Anweisungsbeispiele

Prec ON Hohe Verfahrweggenauigkeit aktiviert
Prec OFF Hohe Verfahrweggenauigkeit deaktiviert

Programmbeispiel

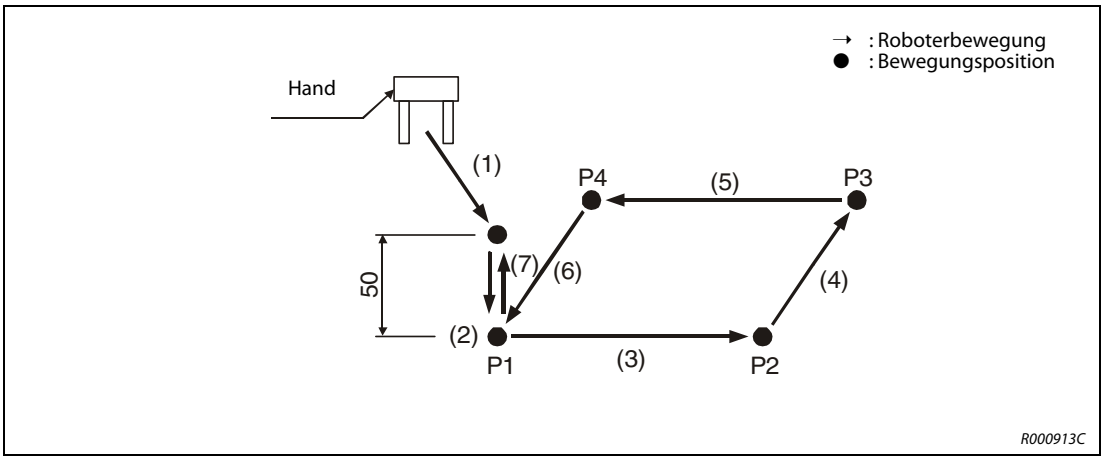


Abb. 4-9: *Verfahrweggenauigkeit*

HINWEIS

Die Ausführung eines Bewegungsbefehls mit hoher Verfahrweggenauigkeit (Prec ON) erhöht die Verfahrwegtreue an der Handspitze des Roboters. Dadurch nehmen jedoch die Beschleunigungs- und Bremszeiten und somit auch die Zykluszeiten zu.
Wird eine weitere Erhöhung der Verfahrweggenauigkeit gewünscht, sollte auf die Verwendung des Cnt-Befehls zur Ausführung kontinuierlicher Roboterbewegungen verzichtet werden.

```
1  Mvs P1, -50      '(1) Position mit Gelenk-Interpolation anfahren, die 50 mm in
                    Werkzeuglängsrichtung von der Position P1 entfernt ist
                    (siehe Achtungshinweis)
2  Ovrd 50          'Legt die Geschwindigkeit für das gesamte Programm auf den
                    halben Maximalwert fest
3  Mvs P1           '(2) Position P1 mittels Linear-Interpolation anfahren
4  Prec ON          'Aktiviert die hohe Verfahrweggenauigkeit
5  Mvs P2           '(3) Verfahrweg zwischen Position P1 und Position P2 mit
                    hoher Genauigkeit zurücklegen
6  Mvs P3           '(4) Verfahrweg zwischen Position P2 und Position P3 mit
                    hoher Genauigkeit zurücklegen
7  Mvs P4           '(5) Verfahrweg zwischen Position P3 und Position P4 mit
                    hoher Genauigkeit zurücklegen
8  Mvs P1           '(6) Verfahrweg zwischen Position P4 und Position P1 mit
                    hoher Genauigkeit zurücklegen
9  Prec OFF         'Deaktiviert die hohe Verfahrweggenauigkeit
10 Mvs P1, -50      '(7) Position mittels Linear-Interpolation anfahren, die 50 mm in
                    Werkzeuglängsrichtung von der Position P3 entfernt ist
                    (siehe Achtungshinweis)
11 End              'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

4.3.8 Hand- und Werkzeugsteuerung

Der Handgreiferzustand (offen/geschlossen) und die Werkzeugdaten können festgelegt werden.

Erläuterung

Befehl	Beschreibung
HOpen	Die festgelegte Hand wird geöffnet.
HClose	Die festgelegte Hand wird geschlossen.
Tool	Die Werkzeugdaten und der Überwachungspunkt können eingestellt werden.

Anweisungsbeispiele

HOpen 1	Öffnet Hand 1
HOpen 2	Öffnet Hand 2
HClose 1	Schließt Hand 1
HClose 2	Schließt Hand 2
Tool (0, 0, 95, 0 , 0, 0)	Als Überwachungspunkt wird ein Punkt definiert, der 95 mm in Werkzeuglängsrichtung vom Handflansch entfernt ist.

Programmbeispiel

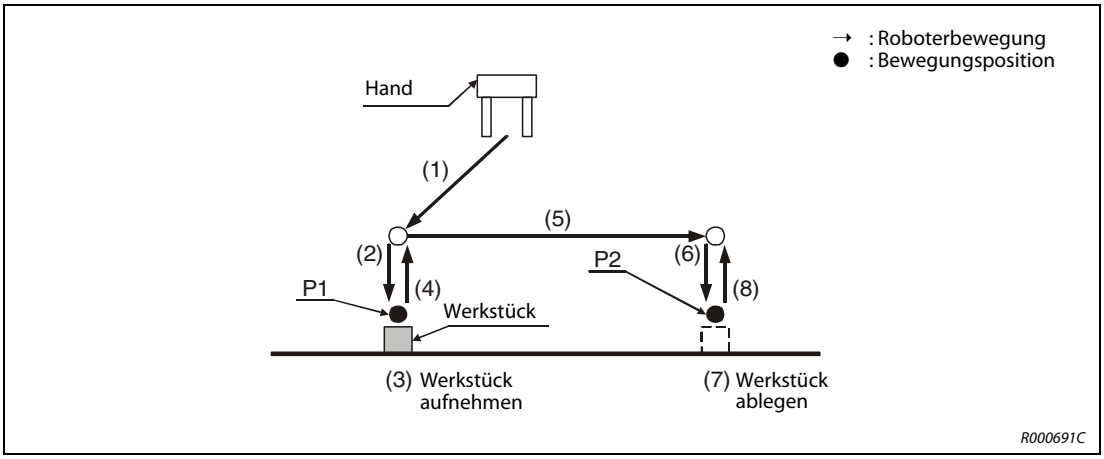


Abb. 4-10: *Verfahrbewegung und Handsteuerung*

```
1  Tool (0, 0, 95, 0, 0, 0)(0, 0)   'Legt die Werkzeuglänge auf 95 mm fest
2  Mov P1, -50                      '(1) Position mittels Gelenk-Interpolation anfahren,
                                    die 50 mm in Werkzeuglängsrichtung von der
                                    Position P1 entfernt ist (siehe Achtungshinweis)
3  Ovrd 50                          'Legt die Geschwindigkeit auf den halben
                                    Maximalwert fest
4  Mvs P1                           '(2) Position P1 mittels Linear-Interpolation anfahren
                                    (Anfahren der Position zur Werkstückaufnahme)
5  HClose 1                         '(3) Schließt Hand 1 (Werkstück aufnehmen)
6  Dly 0.5                          'Wartezeit von 0,5 s
7  Ovrd 100                         'Legt die Geschwindigkeit auf den Maximalwert fest
8  Mvs, -50                         '(4) Position mittels Linear-Interpolation anfahren, die
                                    50 mm in Werkzeuglängsrichtung von der aktuellen
                                    Position (P1) entfernt ist (Anheben des Werkstücks)
                                    (siehe Achtungshinweis)
9  Mov P2, -50                      '(5) Position mittels Gelenk-Interpolation anfahren,
                                    die 50 mm in Werkzeuglängsrichtung von der
                                    Position P2 entfernt ist (siehe Achtungshinweis)
10 Ovrd 50                          'Legt die Geschwindigkeit auf den halben
                                    Maximalwert fest
11 Mvs P2                           '(6) Position P2 mittels Linear-Interpolation anfahren
                                    (Anfahren der Position zur Werkstückablage)
12 HOpen 1                          'Öffnet Hand 1 (Werkstück ablegen)
13 Dly 0.5                          'Wartezeit von 0,5 s
14 Ovrd 100                         '(7) Legt die Geschwindigkeit auf den Maximalwert fest
15 Mvs, -50                         '(8) Position mittels Linear-Interpolation anfahren, die
                                    50 mm in Werkzeuglängsrichtung von der aktuellen
                                    Position (P2) entfernt ist (Entfernen vom Werkstück)
                                    (siehe Achtungshinweis)
16 End                              Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Angehängte Anweisung ⇒ Abschn. 6.3.82

4.4 Palettierung

Mit Hilfe der Palettierungsfunktion können Werkstücke geordnet abgelegt oder geordnete Werkstücke aufgenommen werden. Dabei reicht ein Teachen der Position des Referenz-Werkstücks aus. Alle anderen Positionen werden daraus berechnet.

Erläuterung

Befehl	Beschreibung
Def Plt	Definiert eine Palette
Plt	Berechnet die Koordinaten eines Gitterpunktes der festgelegten Palette und weist die berechneten Koordinaten der festgelegten Position zu

Anweisungsbeispiele

Def Plt 1, P1, P2, P3, P4, 4, 3, 1	Definiert Palette Nummer 1 mit Bezugsposition = P1, Endpunkt A = P2, Endpunkt B = P3, Palletteneckpunkt, der gegenüber der Bezugsposition liegt = P4, Anzahl der Gitterpunkte: 12 (Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A = 4, Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B = 3) und einer Bewegungsrichtung = 1
Def Plt 2, P1, P2, P3, , 8, 5, 2	Definiert Palette Nummer 2 mit Bezugsposition = P1, Endpunkt A = P2, Endpunkt B = P3, Anzahl der Gitterpunkte: 40 (Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A = 5, Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B = 8) und einer Bewegungsrichtung = 2
Def Plt 3, P1, P2, P3, , 8, 5, 3	Definiert kreisförmige Palette Nummer 3 mit 5 Positionen auf einem Kreisbogen über Startposition = P1, Zwischenposition = P2 und Endposition = P3 (insgesamt 3 Punkte)
(Plt 1, 5)	Berechnet die 5te Position der Palette Nummer 1
(Plt 1, M1)	Berechnet die in der numerischen Variablen M1 festgelegte Position der Palette Nummer 1

Palettendefinition und Bewegungsrichtung

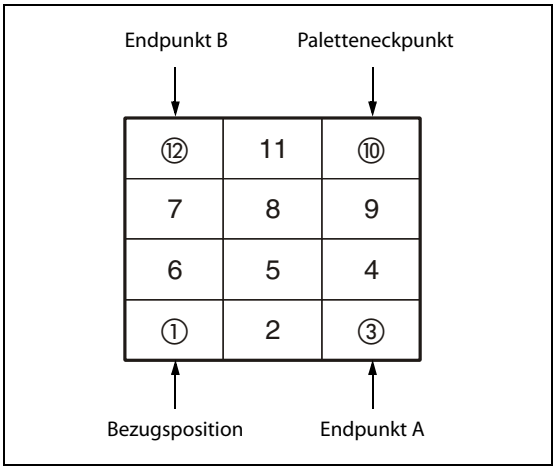


Abb. 4-11:
Palettendefinition mit Bewegungsrichtung = 1 (zickzack)

R000693C

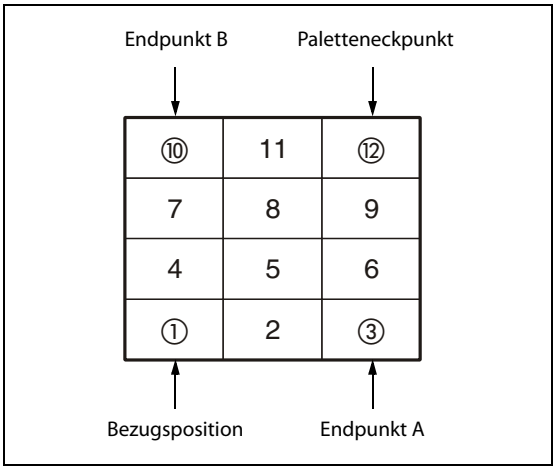


Abb. 4-12:
Palettendefinition mit Bewegungsrichtung = 2 (Richtung beibehalten)

R000694C

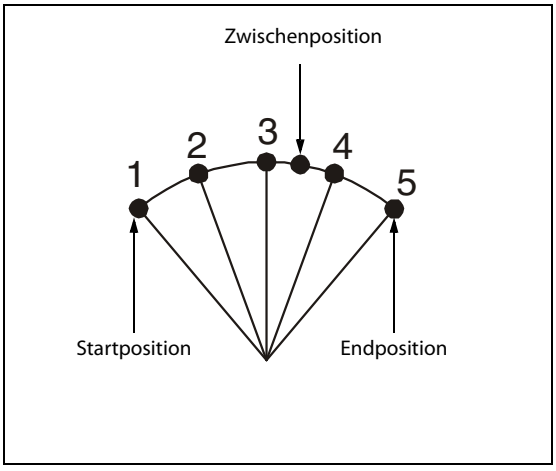


Abb. 4-13:
Palettendefinition mit Bewegungsrichtung = 3 (kreisförmig)

R000695C

HINWEISE

Nimmt eine der Orientierungsdaten A, B oder C den Wert 180° an, so kann dieser Wert bei gleicher Stellung +180° oder −180° entsprechen. Die Uneindeutigkeit des Vorzeichens kann zu internen Verarbeitungsfehlern führen.
Wird eine solche Position bei der Festlegung der Palette als Bezugsposition, Endpunkt A oder B oder Palletteneckpunkt verwendet, so kann die Hand rotieren und sich in unvorhersehbarer Weise bewegen, da die Palettierungsfunktion ihre Positionen durch eine Unterteilung der Distanz zwischen −180° und +180° berechnet.
Ein und dieselbe Position kann über einen Drehwinkel von +180° oder −180° erreicht werden. Möchten Sie für die Palettierungsfunktion Positiondaten verwenden, deren Orientierungsdaten 180° betragen, so versehen sie Sie mit einem Vorzeichen.
Ein ähnliches Phänomen tritt auch auf, wenn bei Positionen, deren Drehwinkel beinahe +/−180° betragen (z.B. +179° oder −179°), unterschiedliche Vorzeichen verwendet werden. Addieren/subtrahieren Sie in diesem Fall 360° zu/von den Orientierungsdaten und korrigieren Sie die Werte so, dass die Vorzeichen übereinstimmen. (Beispiel: Ändern Sie das Vorzeichen von −179° nach +, indem Sie 360° hinzuaddieren. Sie erhalten so den korrigierten Wert von +181°.)
Das Programmbeispiel 1 zeigt einen Fall, bei dem die Orientierungsdaten der Endpunkte (P3 und P4) und der Palletteneckpunkt (P5) in Bezug auf die Bezugsposition (P2) korrigiert werden. Die Handausrichtung bei allen Gitterpunkten der Palette ist dieselbe (Werte A, B und C stimmen überein).
Im Programmbeispiel 2 werden die Werte so korrigiert, dass deren Vorzeichen mit dem der Bezugsposition (P2) übereinstimmen, wenn die Orientierungsdaten bei der Definition der Palette +/−180° erreichen und die Werte der C-Achse für die Endpunkte (P3 und P4) und den Palletteneckpunkt (P5) entweder kleiner als −179° oder größer als +179° ist (Schrittnummer 1 bis 10). (Der Schwellwert zur Korrektur der Daten ist +/−178°.)
Verwenden Sie die Programmbeispiele als Grundlage für Anwendungen, in denen keine hohe Palettierungsgenauigkeit gefordert ist und die Handausrichtung geringfügig korrigiert werden muss.

Aus dem Wert für den Bezugspunkt der Palettendefinition werden durch den Palettierungsbefehl (Plt) die Stellungsmerker (FL1 der Positionsdaten) für die Gitterpunkte berechnet. Werden für die verschiedenen Punkte der Palettendefinition Positionsdaten mit unterschiedlichen Stellungsmerkern verwendet, so kann die Palettierung nicht ausgeführt werden.
Bei der Palettendefinition müssen die Stellungsdaten für die Bezugsposition, die Endpunkte A und B und den Palletteneckpunkt übereinstimmen. Aus der Bezugsposition werden auch die Werte die Multirotationsdaten für die Gitterpunkte abgeleitet (FL2 der Positionsdaten).
Werden für die verschiedenen Punkte der Palettendefinition Positionsdaten mit unterschiedlichen Stellungsmerkern für die Multirotationsdaten verwendet, so kann die Hand rotieren und sich in Abhängigkeit der Roboterstellungen bei der Palettierung und der Interpolationsbefehle (Gelenk-Interpolation, Linear-Interpolation usw.) in unvorhersehbarer Weise bewegen. Wählen Sie in einem solchen Fall die direkte oder indirekte Anfahrt mit Hilfe der Type-Funktion und stellen Sie so sicher, dass sich die Hand in der gewünschten Weise bewegt.

Programmbeispiele

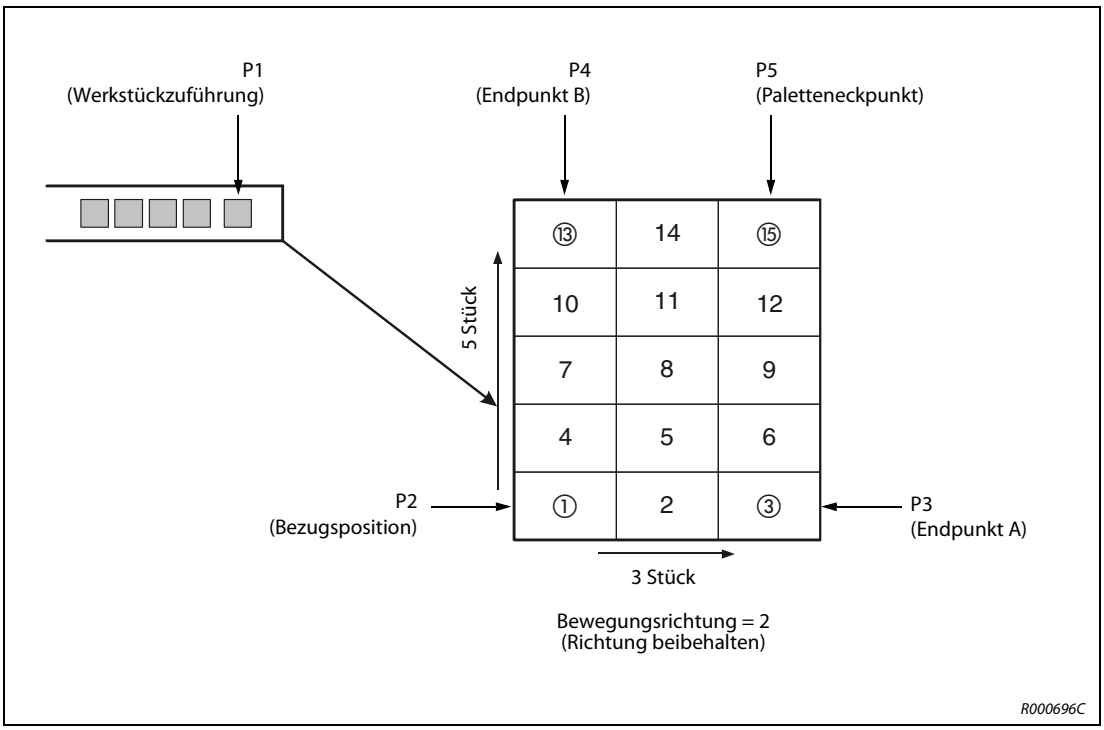


Abb. 4-14: *Palettierung*

Programmbeispiel 1

Die Handausrichtung ist in allen Gitterpunkten der Palette dieselbe (Werte A, B und C stimmen überein).

```
1  P3.A=P2.A                    'Weist dem Orientierungsdatenelement (A) von P3 das
                                Orientierungsdatenelement (A) von P2 zu.
2  P3.B=P2.B                    'Weist dem Orientierungsdatenelement (B) von P3 das
                                Orientierungsdatenelement (B) von P2 zu.
3  P3.C=P2.C                    'Weist dem Orientierungsdatenelement (C) von P3 das
                                Orientierungsdatenelement (C) von P2 zu.
4  P4.A=P2.A                    'Weist dem Orientierungsdatenelement (A) von P4 das
                                Orientierungsdatenelement (A) von P2 zu.
5  P4.B=P2.B                    'Weist dem Orientierungsdatenelement (B) von P4 das
                                Orientierungsdatenelement (B) von P2 zu.
6  P4.C=P2.C                    'Weist dem Orientierungsdatenelement (C) von P4 das
                                Orientierungsdatenelement (C) von P2 zu.
7  P5.A=P2.A                    'Weist dem Orientierungsdatenelement (A) von P5 das
                                Orientierungsdatenelement (A) von P2 zu.
8  P5.B=P2.B                    'Weist dem Orientierungsdatenelement (B) von P5 das
                                Orientierungsdatenelement (B) von P2 zu.
9  P5.C=P2.C                    'Weist dem Orientierungsdatenelement (C) von P5 das
                                Orientierungsdatenelement (C) von P2 zu.
10 Def Plt 1, P2, P3, P4, P5, 3, 5, 2
                                'Definiert Palette Nummer 1 mit Bezugsposition = P2,
                                Endpunkt A = P3, Endpunkt B = P4, Palletteneckpunkt,
                                der gegenüber der Bezugsposition liegt = P5, Anzahl
                                der Gitterpunkte: 15 (Anzahl der Gitterpunkte zwischen
                                Bezugsposition und Endpunkt A = 3, Anzahl der Gitter-
                                punkte zwischen Bezugsposition und Endpunkt B = 5)
                                und einer Bewegungsrichtung = 2
11 M1 = 1                       'Setzt M1 auf „1“ (M1 dient als Zähler)
12 *LOOP                        'Sprungmarke LOOP festgelegt
```

```
13 Mov P1, −50          'Position mittels Gelenk-Interpolation anfahren, die
                        50 mm in Werkzeuglängsrichtung von der Position 1
                        entfernt ist (siehe Achtungshinweis)
14 Ovrd 50              'Legt die Geschwindigkeit auf den halben
                        Maximalwert fest
15 Mvs P1               'Position 1 mittels Linear-Interpolation anfahren
                        (anfahren der Position zur Werkstückaufnahme)
16 HClose 1             'Schließt Hand 1 (Werkstück aufnehmen)
17 Dly 0.5              'Wartezeit von 0,5 s
18 Ovrd 100             'Legt die Geschwindigkeit auf den Maximalwert fest
19 Mvs, −50             'Position mittels Linear-Interpolation anfahren, die
                        50 mm in Werkzeuglängsrichtung von der
                        aktuellen Position (P1) entfernt ist (Anheben des
                        Werkstücks) (siehe Achtungshinweis)
20 P10 = (Plt 1, M1)    'Berechnet die in der numerischen Variablen M1
                        festgelegte Position der Palette Nummer 1 und
                        schreibt den Wert in P10
21 Mov P10, −50         'Position mittels Gelenk-Interpolation anfahren,
                        die 50 mm in Werkzeuglängsrichtung von der
                        Position P10 entfernt ist (siehe Achtungshinweis)
22 Ovrd 50              'Legt die Geschwindigkeit auf den halben
                        Maximalwert fest
23 Mvs P10              'Position P10 mittels Linear-Interpolation anfahren
                        (Anfahren der Position zur Werkstückablage)
24 HOpen 1              'Öffnet Hand 1 (Werkstück ablegen)
25 Dly 0.5              'Wartezeit von 0,5 s
26 Ovrd 100             'Legt die Geschwindigkeit auf den Maximalwert fest
27 Mvs, −50             'Position mittels Linear-Interpolation anfahren,
                        die 50 mm in Werkzeuglängsrichtung von der
                        aktuellen Position (P10) entfernt ist (Entfernen
                        vom Werkstück) (siehe Achtungshinweis)
28 M1 = M1 + 1          'Numerische Variable M1 um 1 erhöhen
                        (Palettenzähler erhöhen)
29 If M1 <= 15 Then *LOOP 'Ist der Wert der numerischen Variablen M1
                        kleiner als 15, springe zur Marke LOOP, sonst
                        gehe in die nächste Zeile.
30 End                  'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Programmbeispiel 2

Korrektur, wenn die Orientierungsdaten +/−180° erreichen

```
1  If Deg(P2.C) < 0 Then GoTo *MINUS                  'Prüft das Vorzeichen des Orientierungs-
                                                      datenelements C von P2 und springt
                                                      zur Marke MINUS, falls es negativ ist
2  If Deg(P3.C) < −178 Then P3.C = P3.C + Rad(+360)   'Addiert 360° zum Orientierungs-
                                                      datenelement C von P3, wenn das
                                                      Element −180° erreicht, um einen
                                                      positiven Wert zu erhalten
3  If Deg(P4.C) < −178 Then P4.C = P4.C + Rad(+360)   'Addiert 360° zum Orientierungs-
                                                      datenelement C von P4, wenn das
                                                      Element −180° erreicht, um einen
                                                      positiven Wert zu erhalten
4  If Deg(P5.C) < −178 Then P5.C = P5.C + Rad(+360)   'Addiert 360° zum Orientierungs-
                                                      datenelement C von P5, wenn das
                                                      Element −180° erreicht, um einen
                                                      positiven Wert zu erhalten
5  GoTo *DEFINE                                       'Sprung zum Unterprogramm DEFINE
6  *MINUS                                             'Sprungmarke MINUS festgelegt
7  If Deg(P3.C) < +178 Then P3.C = P3.C − Rad(+360)   'Addiert 360° zum Orientierungs-
                                                      datenelement C von P3, wenn das
                                                      Element +180° erreicht, um einen
                                                      negativen Wert zu erhalten
8  If Deg(P4.C) < +178 Then P4.C = P4.C − Rad(+360)   'Addiert 360° zum Orientierungs-
                                                      datenelement C von P4, wenn das
                                                      Element +180° erreicht, um einen
                                                      negativen Wert zu erhalten
9  If Deg(P5.C) < +178 Then P5.C = P5.C − Rad(+360)   'Addiert 360° zum Orientierungs-
                                                      datenelement C von P5, wenn das
                                                      Element +180° erreicht, um einen
                                                      negativen Wert zu erhalten
10 *DEFINE                                            'Sprungmarke DEFINE festgelegt
11 Def Plt 1, P2, P3, P4, P5, 3, 5, 2                 'Definiert Palette Nummer 1 mit
                                                      Bezugsposition = P2, Endpunkt A = P3,
                                                      Endpunkt B = P4, Palletteneckpunkt, der
                                                      gegenüber der Bezugsposition liegt = P5,
                                                      Anzahl der Gitterpunkte: 15 (Anzahl der
                                                      Gitterpunkte zwischen Bezugsposition
                                                      und Endpunkt A = 3, Anzahl der Gitter-
                                                      punkte zwischen Bezugsposition und
                                                      Endpunkt B = 5) und einer Bewegungs-
                                                      richtung = 2
12 M1 = 1                                             'Setzt M1 auf „1“ (M1 dient als Zähler)
13 *LOOP                                              'Sprungmarke LOOP festgelegt
14 Mov P1, −50                                        'Position mittels Gelenk-Interpolation
                                                      anfahren, die 50 mm in Werkzeuglängs-
                                                      richtung von der Position 1 entfernt ist
                                                      (siehe Achtungshinweis)
15 Ovrd 50                                            'Legt die Geschwindigkeit auf den halben
                                                      Maximalwert fest
16 Mvs P1                                             'Position 1 mittels Linear-Interpolation
                                                      anfahren (anfahren der Position zur
                                                      Werkstückaufnahme)
17 HClose 1                                           'Schließt Hand 1 (Werkstück aufnehmen)
18 Dly 0.5                                            'Wartezeit von 0,5 s
19 Ovrd 100                                           'Legt die Geschwindigkeit auf den
                                                      Maximalwert fest
```

```
20 Mvs, –50          'Position mittels Linear-Interpolation
                     anfahren, die 50 mm in Werkzeuglängs-
                     richtung von der aktuellen Position (P1)
                     entfernt ist (Anheben des Werkstücks)
                     (siehe Achtungshinweis)
21 P10 = (Plt 1, M1) 'Berechnet die in der numerischen
                     Variablen M1 festgelegte Position der
                     Palette Nummer 1 und schreibt den
                     Wert in P10
22 Mov P10, –50      'Position mittels Gelenk-Interpolation
                     anfahren, die 50 mm in Werkzeuglängs-
                     richtung von der Position P10 entfernt ist
                     (siehe Achtungshinweis)
23 Ovrd 50           'Legt die Geschwindigkeit auf den halben
                     Maximalwert fest
24 Mvs P10           'Position P10 mittels Linear-Interpolation
                     anfahren (Anfahren der Position zur
                     Werkstückablage)
25 HOpen 1           'Öffnet Hand 1 (Werkstück ablegen)
26 Dly 0.5           'Wartezeit von 0,5 s
27 Ovrd 100          'Legt die Geschwindigkeit auf den
                     Maximalwert fest
28 Mvs, –50          'Position mittels Linear-Interpolation
                     anfahren, die 50 mm in Werkzeuglängs-
                     richtung von der aktuellen Position (P10)
                     entfernt ist (Entfernen vom Werkstück)
                     (siehe Achtungshinweis)
29 M1 = M1 + 1       'Numerische Variable M1 um 1 erhöhen
                     (Palettenzähler erhöhen)
30 If M1 <= 15 Then *LOOP 'Ist der Wert der numerischen Variablen
                     M1 kleiner als 15, springe zur Marke LOOP,
                     sonst gehe in die nächste Zeile.
31 End               'Programmende
```

ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Grenzen der Palettierungsdefinition

Bei einigen Robotermodellen (RV-2SQ usw.), deren J1- oder J4-Achse einen Drehwinkel von +/−180° überschreiten kann, ist die Definition einer Palette, bei der dieser Drehwinkel überschritten wird, nicht möglich. Wird eine solche Position definiert, erfolgt eine Fehlermeldung. Abb. 4-15 (A) und Abb. 4-16 (A) zeigen Beispiele dazu.
Teilen Sie in einem solchen Fall die Palette bei der Definition auf (siehe Abb. 4-15 (B) und Abb. 4-16 (B)):
Palette 1: Der Drehwinkel der J1-Achse beträgt 0° bis +180° oder 0° bis –180°.
Palette 2: Der Drehwinkel der J1-Achse beträgt +180° bis +240° oder –180° bis –240°.

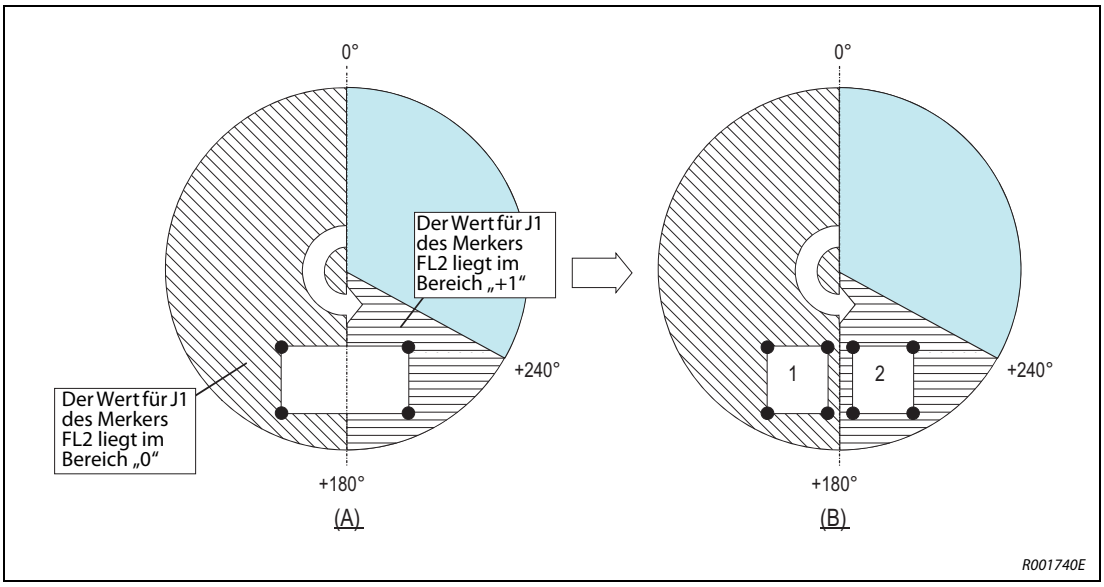


Abb. 4-15: *Grenzen der Palettierungsfunktion (1)*

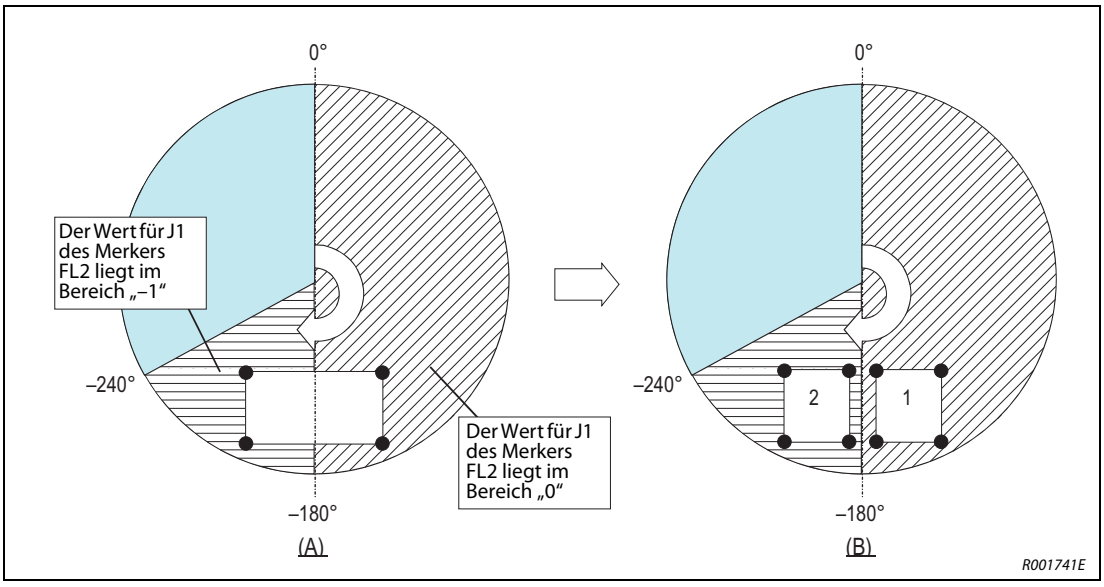


Abb. 4-16: *Grenzen der Palettierungsfunktion (2)*

Befehl steht in Beziehung zu folgenden Funktionen:

Ausdrücke und Operationen	⇒	Abschn. 4.8
Verzweigung und Wartezeit	⇒	Abschn. 4.5.1

4.5 Programmsteuerung

Der Programmfluss kann über Verzweigungen, Interrupts, Unterprogrammaufrufe, Stoppbefehle usw. gesteuert werden.

4.5.1 Verzweigungen und Wartezeit

Ein Sprung zu einem bestimmten Programmschritt kann durch eine unbedingte oder durch eine bedingte Verzweigung erfolgen.

Erläuterung

Befehl	Beschreibung
GoTo	Bewirkt einen unbedingten Sprung zu einer Marke
On GoTo	Bewirkt einen Sprung in Abhängigkeit vom Wert einer Variablen Die Reihenfolge der Sprungziele entspricht der der Integer-Zahlenreihe (0, 1, 2, 3, 4 ...).
If Then Else (Anweisungen in einer Zeile)	Bewirkt einen Sprung in Abhängigkeit vom Wert einer Variablen und den festgelegten Bedingungen für diesen Wert Die Bedingungen für die Werte können frei gewählt werden. Es darf nur eine Verzweigungsart pro Anweisung verwendet werden. Ist die Bedingung erfüllt, wird die Then-Anweisung ausgeführt. Ist die Bedingung nicht erfüllt, wird die Else-Anweisung ausgeführt.

If Then
Else
End If
(Anweisungen in mehreren Zeilen)
	Bewirkt die Abarbeitung von einem oder von mehren Programmschritten in Abhängigkeit vom Wert einer Variablen und den festgelegten Bedingungen für diesen Wert
	Die Bedingungen für die Werte können frei gewählt werden. Es darf nur eine Verzweigungsart pro Anweisung verwendet werden. Ist die Bedingung erfüllt, werden die Zeilen zwischen der Then- und der Else-Anweisung ausgeführt. Ist die Bedingung nicht erfüllt, werden die Zeilen zwischen der Else- und der End If-Anweisung ausgeführt.

Select Case End Select	Bewirkt einen Sprung in Abhängigkeit vom Wert einer Variablen und den festgelegten Bedingungen für diesen Wert Die Bedingungen für die Werte können frei gewählt werden. Es dürfen mehrere Verzweigungsarten pro Anweisung verwendet werden.
Break	Bewirkt einen Sprung zu dem Programmschritt nach der End Select-Anweisung
Wait	Bewirkt eine Wartezeit, bis eine Variable den festgelegten Wert erreicht hat

Anweisungsbeispiele

GoTo *FN	Unbedingter Sprung zur Marke FN
On M1 GoTo *L1, *L2, *L3	Sprung zur Marke L1, falls der Wert der Variablen M1 = 1 ist, Sprung zur Marke L2, falls M1 = 2 und Sprung zur Marke L3, falls M1 = 3 Entspricht M1 keinem dieser Werte, wird der nächste Programmschritt ausgeführt.
If M1 = 1 Then *L1	Sprung zur Marke L1, falls M1 = 1, sonst wird das Programm mit dem nächsten Programmschritt fortgesetzt
If M1 = 1 Then *L1 Else *L2	Sprung zur Marke L1, falls M1 = 1, sonst Sprung zur Marke L2

If M1 = 1 Then
  M2 = 1
  M3 = 2
Else
  M2 = −1
  M3 = −2
EndIf
	Ist M1 = 1, werden die Anweisungen M2 = 1 und M3 = 2
	ausgeführt. Ist M1 ≠ 1, werden die Anweisungen
	M2 = −1 und M3 = −2 ausgeführt.

Select M1
  Case 10
  :
  Break
  Case IS 11
  :
  Break
  Case IS < 5
  :
  Break
  Case 6 TO 9
  :
  Break
  Default
  :
End Select
	Sprung zur Case-Anweisung in Abhängigkeit von M1
	Ist M1 = 10, wird das Programm nur zwischen
	Case 10 und Case IS 11 ausgeführt.
	Sprung zum Programmschritt nach der End Select-Anweisung
	Ist M1 = 11, wird das Programm nur zwischen den Anweisungen
	Case IS 11 und Case IS < 5 ausgeführt.
	Sprung zum Programmschritt nach der End Select-Anweisung
	Ist M1 < 5, wird das Programm nur zwischen
	Case IS < 5 und Case 6 TO 9 ausgeführt.
	Sprung zum Programmschritt nach der End Select-Anweisung
	Ist 6 < M1 < 9, wird das Programm nur zwischen
	Case 6 TO 9 und Default ausgeführt.
	Sprung zum Programmschritt nach der End Select-Anweisung
	Entspricht M1 keinem der Werte, wird das Programm
	nur zwischen Default und End Select ausgeführt.

Wait M_In(1) = 1	Wartezeit, bis Eingangsbit 1 eingeschaltet wird

Befehl steht in Beziehung zu folgenden Funktionen:

Programmschleife	⇒	Abschn. 4.5.2
Interrupt	⇒	Abschn. 4.5.3
Unterprogramm	⇒	Abschn. 4.5.4
Eingangssignale	⇒	Abschn. 4.6.1

4.5.2 Programmschleife

Bestimmte Programmteile können in Abhängigkeit einer Bedingung wiederholt werden.

Erläuterung

Befehl	Beschreibung
For Next	Bewirkt eine Wiederholung des Programmteils zwischen der For- und Next-Anweisung, bis die Abbruchbedingung erfüllt ist
While Wend	Bewirkt eine Wiederholung des Programmteils zwischen der While- und Wend-Anweisung, solange die Ausführungsbedingung erfüllt ist

Anweisungsbeispiele

```
For M1 = 1 To 10                    10-malige Wiederholung des Programmteils
   :                                zwischen der For- und Next-Anweisung
   :                                Der Startwert der Variablen M1 ist 1. Er wird bei
Next                                jeder Wiederholung um 1 erhöht.

For M1 = 0 To 10 Step 2             6-malige Wiederholung des Programmteils
   :                                zwischen der For- und Next-Anweisung
   :                                Der Startwert der Variablen M1 ist 0. Er wird bei
Next                                jeder Wiederholung um 2 erhöht.

While (M1 >= 1) And (M1 <= 10)      Wiederholung des Programmteils zwischen der
   :                                While- und Wend-Anweisung, solange der Wert
   :                                der numerischen Variablen M1 größer oder gleich 1 und
WEnd                                kleiner oder gleich10 ist
```

Befehl steht in Beziehung zu folgenden Funktionen:

Verzweigung ⇒ Abschn. 4.5.1
Interrupt ⇒ Abschn. 4.5.3
Eingangssignale ⇒ Abschn. 4.6.1

4.5.3 Interrupt

Die Ausführung eines Programms kann mittels eines Interrupts unterbrochen und verzweigt werden.

Erläuterung

Befehl	Beschreibung
Def Act	Festlegung des Status und der Ausführung des Interrupts
Act	Freigeben oder Sperren eines Interrupts
Return	Bewirkt den Rücksprung aus einer Interrupt-Routine zu dem Programmschritt, in dem der Interrupt aufgerufen wurde

Anweisungsbeispiele

Def Act 1, M_In(10) = 1 GoSub *SUB1	Definiert einen Unterprogrammsprung zur Marke SUB1 mit der Priorität 1, falls das Eingangssignalbit 10 auf „1“ gesetzt wird und der Roboter bis zum Stillstand abgebremst ist. Die Bremszeit hängt von den über die Accel- und Ovrd-Anweisungen festgelegten Werten ab.
Def Act 2, M_In(11) = 1 GoSub *SUB2, L	Definiert einen Unterprogrammsprung zur Marke SUB2 mit der Priorität 2, falls das Eingangssignalbit 11 auf „1“ gesetzt wird und die Ausführung der aktuellen Anweisung beendet ist
Def Act 3, M_In(12) = 1 GoSub *SUB3, S	Definiert einen Unterprogrammsprung zur Marke SUB3 mit der Priorität 3, falls das Eingangssignalbit 12 auf „1“ gesetzt wird und der Roboter in der kürzestmöglichen Zeit mit dem kürzesten Bremsweg bis zum Stillstand abgebremst ist.
Act 1 = 1	Interrupt 1 freigeben
Act 2 = 0	Interrupt 2 sperren
Return 0	Rücksprung zu dem Programmschritt, in dem der Interrupt aufgerufen wurde
Return 1	Rücksprung zu dem Programmschritt, der dem Schritt mit dem Interrupt-Aufruf folgt

Befehl steht in Beziehung zu folgenden Funktionen:

Verzweigung	⇒	Abschn. 4.5.1
Unterprogramm	⇒	Abschn. 4.5.4
Kommunikation	⇒	Abschn. 4.7

4.5.4 Unterprogramm

Mit Hilfe von Unterprogrammen und Routinen kann die Anzahl der Schritte im Hauptprogramm reduziert werden. Ein hierarchischer Aufbau und eine bessere Verständlichkeit des Programms sind somit möglich.

Erläuterung

Befehl	Beschreibung
GoSub	Bewirkt einen Sprung zu einem Unterprogramm, das durch einen festgelegten Programmschritt oder eine Marke definiert ist
On GoSub	Bewirkt einen Sprung zu einem Unterprogramm in Abhängigkeit vom Wert einer Variablen Die Reihenfolge der Sprungziele entspricht der Integer-Zahlenreihe (0, 1, 2, 3, 4 ...).
Return	Bewirkt den Rücksprung aus einer Interrupt-Routine zu dem Programmschritt, der dem Schritt folgt, aus dem der Unterprogrammaufruf mit dem GoSub-Befehl erfolgte
CallP	Bewirkt den Aufruf eines Programms Wird im aufgerufenen Programm die End-Anweisung ausgeführt, erfolgt der Rücksprung zu dem Programmschritt des aufrufenden Programms, der dem Schritt folgt, aus dem der Programmaufruf mit dem CallP-Befehl erfolgte. Beim Programmaufruf können Daten übergeben werden.
FPrm	Legt die Daten fest, die beim Aufruf eines Programms mit dem CALLP-Befehl übergeben werden

Anweisungsbeispiele

GoSub *GET	Springt zum Unterprogramm mit der Marke GET
ON M1 GoSub *L1, *L2, *L3	Springt zum Unterprogramm mit der Marke L1, falls M1 = 1 ist, springt zum Unterprogramm mit der Marke L2, falls M1 = 2 ist und springt zum Unterprogramm mit der Marke L3, falls M1 = 3 ist Entspricht M1 keinem der Werte, wird der nächste Programmschritt ausgeführt.
Return	Rücksprung zu dem Programmschritt, der dem Unterprogrammaufruf mit dem GoSub-Befehl folgt
CallP "10"	Aufruf des Programms Nummer 10
CallP "20", M1, P1	Aufruf des Programms Nummer 20 und Übergabe der numerischen Variablen M1 und der Positionsvariablen P1
FPrm M10, P10	Festlegung der numerischen Variablen M10 und der Positionsvariablen P10, die bei Aufruf des Unterprogramms mit CallP übernommen werden

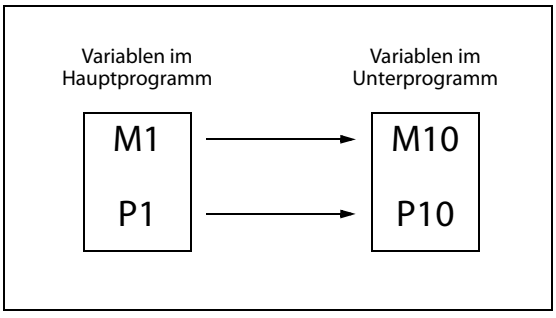


Abb. 4-17:
Übergabe der numerischen Variablen und der Positionsvariablen

Befehl steht in Beziehung zu folgenden Funktionen:

Interrupt	⇒	Abschn. 4.5.3
Kommunikation	⇒	Abschn. 4.7
Verzweigung	⇒	Abschn. 4.5.1

4.5.5 Timer

Die Funktion ermöglicht die Festlegung eines Wartestatus. Bei Verwendung des Dly-Befehls mit einem Impulsausgang kann die Impulsdauer eingestellt werden.

Erläuterung

Befehl	Beschreibung
Dly	Festlegung von Wartezeit und Impulsdauer

Anweisungsbeispiele

Dly 0.05	Wartezeit von 0,05 s
M_Out(10) = 1 Dly 0.5	Ausgangsbit 10 für 0,5 s einschalten

Befehl steht in Beziehung zu folgenden Funktionen:

Impulsausgang ⇒ Abschn. 4.6.2

4.5.6 Stopp

Durch den Hlt-Befehl wird der Programmablauf unterbrochen und der Roboter bis zum Stillstand abgebremst.

Erläuterung

Befehl	Beschreibung
Hlt	Bewirkt eine Unterbrechung des Programmablaufs und ein Stoppen des Roboters Bei einem erneuten Programmstart wird das unterbrochene Programm vom nächsten Programmschritt an ausgeführt.
End	Definiert das Ende eines Programmzyklus Im kontinuierlichen Betrieb erfolgt nach Abarbeitung der End-Anweisung eine erneute Ausführung des Programms ab der Startzeile. Im zyklischen Betrieb endet das Programm bei einem Zyklusstopp nach Abarbeitung der End-Anweisung.

Anweisungsbeispiele

Hlt	Programmablauf unterbrechen und Roboterbewegung stoppen
If M_In(20) = 1 Then Hlt	Programmablauf unterbrechen und Roboterbewegung stoppen, wenn Eingangsbit 20 gleich 1 ist
Mov P1 WthIf M_IN(20) = 1, Hlt	Programmablauf unterbrechen und Roboterbewegung stoppen, wenn bei Anfahrt der Position P1 das Eingangsbit 20 gleich 1 ist Programmablauf wird auch mitten in der Ausführung unterbrochen
End	Beendet das Programm bei Ausführung des End-Befehls bei zyklischer Ausführung oder bei Betätigung der [END]-Taste

Befehl steht in Beziehung zu folgenden Funktionen:

Angehängte Anweisung ⇒ Abschn. 6.3.82

4.6 Ein- und Ausgabe externer Signale

4.6.1 Eingangssignale

Die Steuergeräte können Signale von externen Geräten, z.B. einer SPS, empfangen und verarbeiten. Ein Einlesen der Eingangssignale erfolgt über die Roboterstatusvariablen M_In() usw. (siehe auch Seite 5-18).

Erläuterung

Befehl	Beschreibung
Wait	Bewirkt eine Programmunterbrechung, bis die festgelegte Eingangsbedingung erfüllt ist

Systemvariablen

M_In, M_Inb, M_Inw und M_DIn

Anweisungsbeispiele

Wait M_In(1) = 1	Wartet, bis das Eingangsbit 1 eingeschaltet wird
M1 = M_Inb(20)	Schreibt die Eingangssignalbits 20 bis 27 als Byte (8 Bit) in die numerische Variable M1
M1 = M_Inw(5)	Schreibt die Eingangssignalbits 5 bis 20 als Wort (16-Bit) in die numerische Variable M1

Befehl steht in Beziehung zu folgenden Funktionen:

Ausgangssignale	⇒	Abschn. 4.6.2
Verzweigung	⇒	Abschn. 4.5.1
Interrupt	⇒	Abschn. 4.5.3

4.6.2 Ausgangssignale

Die Steuergeräte können Signale an externe Geräte, z.B. eine SPS, ausgeben. Eine Ausgabe der Ausgangssignale erfolgt über die Roboterstatusvariablen M_Out() usw. (siehe auch Seite 5-18).

Erläuterung

Befehl	Beschreibung
Clr	Bewirkt ein Zurücksetzen der allgemeinen Ausgangssignale auf das über die Parameter ORST0 bis ORST224 vorgegebene Bitmuster

Systemvariablen

M_Out, M_Outb, M_Outw und M_Dout

Anweisungsbeispiele

Clr 1	Zurücksetzen der Ausgänge auf das vorgegebene Bitmuster
M_Out(1) = 1	Ausgangsbit 1 einschalten
M_Outb(8) = 0	8 Bits, von Ausgangsbit 8 bis 15, ausschalten
M_Outw(20) = 0	16 Bits, von Ausgangsbit 20 bis 35, ausschalten
M_Out(1) = 1 Dly 0.5	Ausgangsbit 1 für 0,5 s einschalten (Impulsausgang)
M_Outb(10) = &H0F	4 Bits, von Ausgangsbit 10 bis 13 einschalten und 4 Bits von Ausgangsbit 14 bis 17 ausschalten

Befehl steht in Beziehung zu folgenden Funktionen:

Eingangssignale	⇒	Abschn. 4.6.1
Timer	⇒	Abschn. 4.5.5

4.7 Kommunikation

Die Kommunikationsfunktionen ermöglichen einen Datenaustausch zwischen dem Steuergerät und externen Geräten (z. B. Personalcomputer). Sie stehen bei den Geräten der CRnQ-Serie nicht zur Verfügung.

Erläuterung

Befehl	Beschreibung
Open	Öffnet eine Kommunikationsleitung
Close	Schließt eine geöffnete Kommunikationsleitung
Print #	Ausgabe von Daten im ASCII-Format Nach jeder Print-Anweisung wird ein „Carriage Return“ ausgeführt.
Input #	Eingabe von Daten im ASCII-Format Nach jeder Input-Anweisung wird ein „Carriage Return“ ausgeführt.
On Com GoSub	Legt den Sprung in ein Unterprogramm fest, wenn ein Interrupt von einer Kommunikationsleitung anliegt Ein Interrupt erfolgt durch Eingabe von Daten über ein externes Gerät.
Com On	Gibt die Interrupts von Kommunikationsleitungen frei
Com Off	Sperrt die Interrupts von Kommunikationsleitungen Ein generierter Interrupt bleibt wirkungslos.
Com Stop	Der Interrupt-Prozess von Kommunikationsleitungen wird unterbrochen. Ein generierter Interrupt wird gespeichert und nach Freigabe der Kommunikationsleitung abgearbeitet.

Anweisungsbeispiele

Open "COM1:" AS#1	Öffnet die RS232C-Schnittstelle als Datei Nr. 1
Close #1	Schließt die Datei Nr. 1
Close	Schließt alle geöffneten Dateien
Print #1, "TEST"	Zeichenkette „TEST“ an Schnittstelle ausgeben
Print #2, "M ="; M1	Zeichenkette „M =“ an Datei Nr. 1 und M1 an Datei Nr. 2 ausgeben Ausgabe: „M1 = 1“ + CR (wenn M1 = 1)
Print #3, P1	Positionsvariable P1 an Datei Nr. 3 ausgeben Ausgabe: „(123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)
Print #1, M5, P5	Numerische Variable M5 und Positionsvariable P5 an Datei Nr. 1 ausgeben M5 und P5 werden durch ein Komma getrennt (hexadezimal, 2C). Ausgabe: „8, (123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn M5 = 8, P5: X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)
Input #1, M3	Wandelt die Eingangsdaten in einen Wert um und schreibt diesen in die numerische Variable M3 Eingabe: „8“ + CR (bei Eingabe von 8)
Input #1, P10	Wandelt die Eingangsdaten in Werte um und schreibt diese in die Positionsvariable P10 Eingabe: „(123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn P10: X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)

Input #1, M8, P6	Schreibt die ersten eingegebenen Daten in die numerische Variable M8 Die folgenden Daten werden in die Positionsvariable P6 geschrieben. M8 und P6 werden durch ein Komma getrennt (hexadezimal, 2C). Eingabe: „7, (123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn M8 = 7, P6: X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)
On Com(1) GoSub *SUB3	Springt zum Unterprogramm mit der Marke SUB3, falls über COM1 eine Dateneingabe erfolgt
ON Com(2) GoSub *RECV	Springt zur Marke RECV, falls über COM2 eine Dateneingabe erfolgt
Com(1) On	Freigabe des Interrupts von COM1
Com(2) Off	Sperren des Interrupts von COM2
Com(1) Stop	Stoppt den Interrupt von COM1

Befehl steht in Beziehung zu folgenden Funktionen:

Unterprogramm	⇒	Abschn. 4.5.4
Interrupt	⇒	Abschn. 4.5.3

4.8 Ausdrücke und Operationen

4.8.1 Übersicht

Folgende Tabelle zeigt die in MELFA-BASIC V möglichen Operationen, deren Verwendung und Anwendungsbeispiele:

Operation	Operator	Bedeutung	Beispiel	
Substitution	=	Der rechte Operand wird in den linken geschrieben.	P1 = P2 P5 = P_CURR P10.Z = 100.0 M1 = 1 STS$ = "OK"	Schreibe P2 in P1. Schreibe die aktuelle Position in P5. Setze die Z-Koordinate von P10 auf 100.0. Schreibe den Wert 1 in die numerische Variable M1. Schreibe die Zeichenkette „OK“ in die Zeichenkettenvariable STS.
Operationen mit numerischen Daten	+	Addition	P10 = P1 + P2 Mov P8 + P9 M1 = M1 + 1 STS$ = "ERR" + "001"	Schreibe das Ergebnis der Addition von P1 und P2 in P10. Fahre die Position an, die sich aus der Summe von P8 und P9 ergibt. Addiere 1 zu der numerischen Variablen M1. Addiere die Zeichenketten „ERR“ und „001“ und schreibe das Ergebnis in die Zeichenkettenvariable STS.
	–	Subtraktion	P10 = P1 – P2 Mov P8 – P9 M1 = M1 – 1	Schreibe das Ergebnis der Subtraktion von P1 minus P2 in P10. Fahre die Position an, die sich aus der Differenz von P8 und P9 ergibt. Subtrahiere 1 von der numerischen Variablen M1.
	*	Multiplikation	P1 = P10 * P3 M1 = M1 * 5	Schreibe das Ergebnis der relativen Konvertierung von P10 und P3 in P1. Multipliziere die numerische Variable M1 mit 5.
	/	Division	P1 = P10 / P3 M1 = M1 / 2	Schreibe das Ergebnis der umgekehrten relativen Konvertierung von P10 und P3 in P1. Dividiere die numerische Variable M1 durch 2.
	^	Exponential-Funktion	M1 = M1 ^ 2	Quadriere die numerische Variable M1.
	\	Integer-Division	M1 = M1 \ 3	Dividiere die numerische Variable durch 3 und runde das Ergebnis ab.
	Mod	Modulo-Arithmetik	M1 = M1 Mod 3	Dividiere die numerische Variable M1 durch 3 und schreibe den Rest in M1.
	–	Vorzeichenumkehr	P1 = –P1 M1 = –M1	Kehre das Vorzeichen jeder Koordinate der Positionsvariablen P1 um. Kehre das Vorzeichen der numerischen Variablen M1 um.

Tab. 4-3: *Ausdrücke und Operationen (1)*

Operation	Operator	Bedeutung	Beispiel	
Vergleichs-operation	=	Gleich	If M1 = 1 Then *L1 If STS$ = "OK" Then *L2	Springe zur Marke L1, falls die numerische Variable M1 gleich 1 ist. Springe zur Marke L2, falls die Zeichenkettenvariable STS$ „OK“ ist.
	<> oder ><	Ungleich	If M1 <> 2 Then *L3 If STS$ <> "OK" Then *L4	Springe zur Marke L3, falls die numerische Variable M1 ungleich 2 ist. Springe zur Marke L4, falls die Zeichenkettenvariable STS$ ungleich „OK“ ist.
	<	Kleiner als	If M1 < 10 Then *L3 If STS$ < 3 Then *L4	Springe zur Marke L3, falls die numerische Variable M1 kleiner als 10 ist. Springe zur Marke L4, falls die Anzahl der Zeichen der Zeichenkette STS$ kleiner als 3 ist.
	>	Größer als	If M1 > 9 Then *L3 If STS$ > 2 Then *L4	Springe zur Marke L3, falls die numerische Variable M1 größer als 9 ist. Springe zur Marke L4, falls die Anzahl der Zeichen in der Zeichenkettenvariablen STS$ größer als 2 ist.
	=< oder <=	Kleiner oder gleich	If M1 <= 10 Then *L3 If STS$ <= 5 Then *L4	Springe zur Marke L3, falls die numerische Variable M1 kleiner oder gleich 10 ist. Springe zur Marke L4, falls die Anzahl der Zeichen in der Zeichenkettenvariablen STS$ kleiner oder gleich 5 ist.
	=> oder >=	Größer oder gleich	If M1 => 11 Then *L3 If STS$ >= 6 Then *L4	Springe zur Marke L3, falls die numerische Variable M1 größer oder gleich 11 ist. Springe zur Marke L4, falls die Anzahl der Zeichen in der Zeichenkettenvariablen STS$ größer oder gleich 6 ist.
Logische Operation	And	Logisches UND	M1 = M_Inb(1) And &H0F	Konvertiere die Eingangsbits 1 bis 4 und schreibe das Ergebnis in die numerische Variable M1. (Eingangsbits 5 bis 8 bleiben AUS.)
	Or	Logisches ODER	M_Outb(20) = M1 Or &H80	Ausgabe der numerischen Variablen M1 an Ausgangsbits 20 bis 27 (Ausgangsbit 27 ist dabei immer EIN)
	Not	Negation	M1 = Not M_Inw(1)	Negiere die Eingangsbits 1 bis 16 und schreibe den Wert in die numerische Variable M1.
	Xor	Exklusives ODER	M2 = M1 Xor M_Inw(1)	Schreibe das Ergebnis der exklusiven ODER-Verknüpfung von M1 und den Eingangsbits 1 bis 16 in die numerische Variable M2.
	<<	Logische Links-verschiebung	M1 = M1 << 2	Verschiebe die numerische Variable M1 2 Bits nach links.
	>>	Logische Rechts-verschiebung	M1 = M1 >> 1	Verschiebe die numerische Variable M1 1 Bit nach rechts.

Tab. 4-3: *Ausdrücke und Operationen (2)*

4.8.2 Relative Konvertierung von Positionsdaten

Numerische Variable können mit Hilfe der 4 Grundrechenarten verarbeitet werden. Rechenoperationen mit Positionsvariablen beinhalten darüber hinaus auch eine Konvertierung der Koordinaten. Dies soll an Hand einfacher Beispiele erläutert werden.

Relative Konvertierung (Multiplikation)

Beispiel▽

```
1  P2 = (10,5,0,0,0,0)(0,0)        'Definiert Position 2
2  P100 = P1 * P2                  'Multiplikation von P1 und P2
3  Mov P1                          'Position P1 mittels Gelenk-Interpolation anfahren
4  Mvs P100                        'Position P100 mittels Linear-Interpolation anfahren
```

mit P1 = (200,150,100,0,0,45)(4,0)

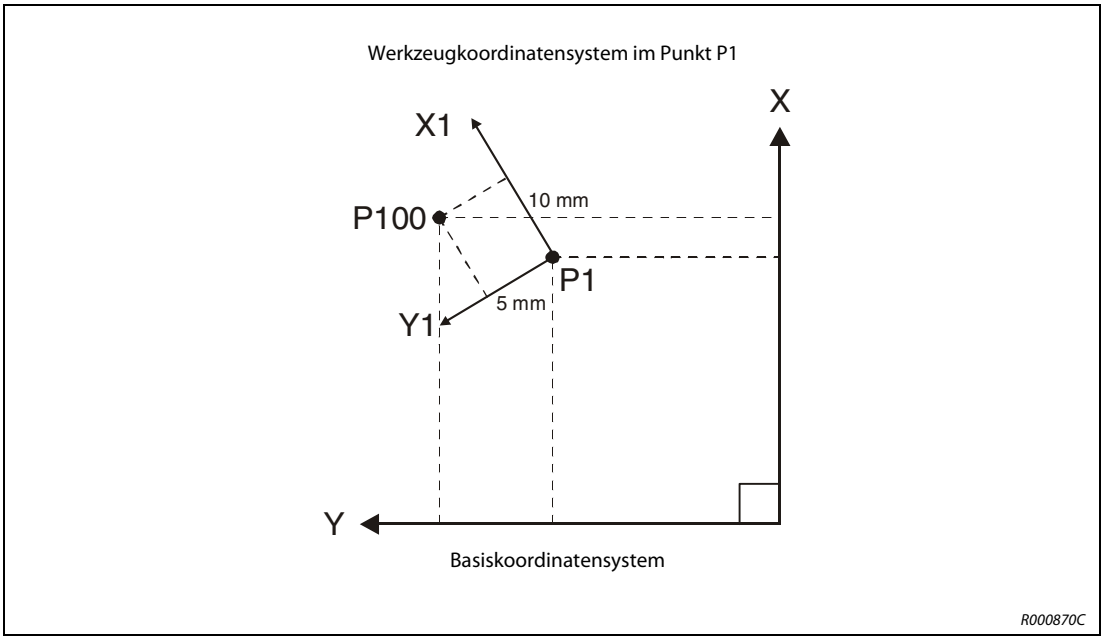


Abb. 4-18: *Multiplikation von Positionsvariablen*

Im oben dargestellten Beispiel erfolgt von der geteachten Position P1 aus eine relative Bewegung der Handspitze im Werkzeugkoordinatensystem. Die X- und Y-Koordinaten der Position P2 entsprechen dabei den im Werkzeugkoordinatensystem zurückgelegten Strecken. Die relative Konvertierung erfolgt durch Multiplikation der Positionsvariablen. Dabei muss beachtet werden, dass ein Vertauschen von Multiplikant und Multiplikator zu einem anderen Ergebnis führt. Die Variable, die die Größe der zurückzulegenden Strecken der relativen Bewegung festlegt (P2), muss als 2ter Faktor eingegeben werden.

Sind die Stellungsdaten der Position P2 (A, B und C) gleich 0, bleibt die Stellung der Position P1 erhalten. Sind die Daten ungleich 0, ergibt sich die neue Stellung bezogen auf die Stellung der Position P1 durch eine relative Drehung der Hand um die Z-, Y- und X-Achse (in der Reihenfolge C, B und A).

Die Multiplikation von Positionsdaten entspricht einer Addition, die Division einer Subtraktion im Werkzeugkoordinatensystem.

△

Relative Konvertierung (Addition)

Beispiel▽

```
1  P2 = (5,10,0,0,0,0)(0,0)      'Definiert Position 2
2  P100 = P1 + P2                'Addition von P1 und P2
3  Mov P1                        'Position P1 mittels Gelenk-Interpolation anfahren
4  Mvs P100                      'Position P100 mittels Linear-Interpolation anfahren
```

mit P1 = (200,150,100,0,0,45)(4,0)

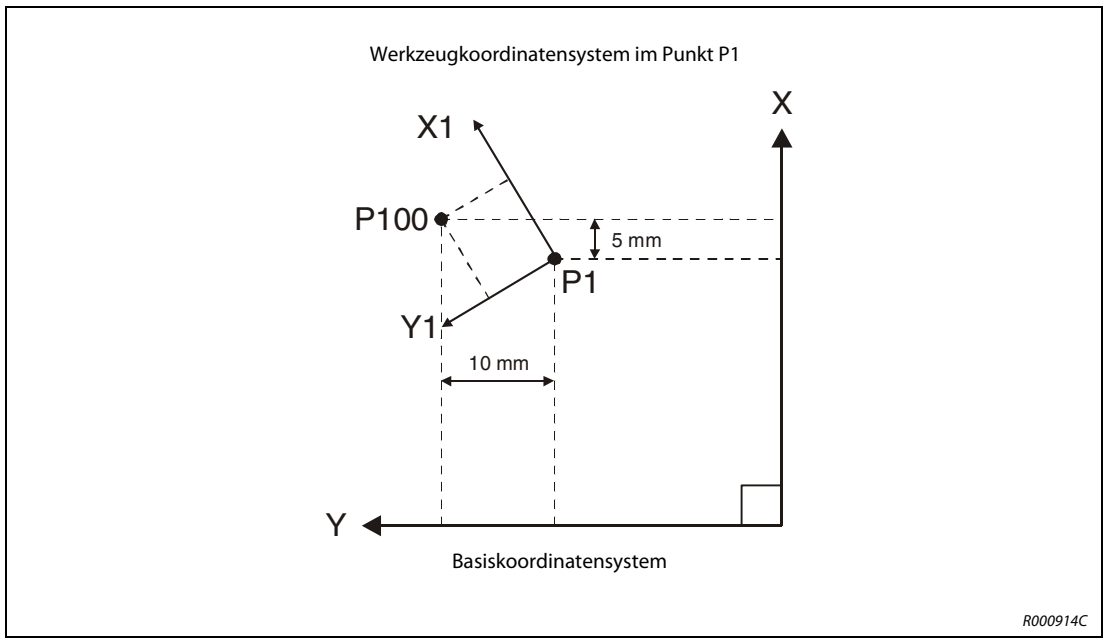


Abb. 4-19: *Addition von Positionsvariablen*

Im oben dargestellten Beispiel erfolgt von der geteachten Position P1 aus eine relative Bewegung der Handspitze im Basiskoordinatensystem. Die X- und Y-Koordinaten der Position P2 entsprechen dabei den im Basiskoordinatensystem zurückgelegten Strecken. Die relative Konvertierung erfolgt durch Addition der Positionsvariablen.

Durch einen Wert für die C-Achse der Position P2 kann der Wert der C-Achse von Position P100 geändert werden. Das Ergebnis entspricht der Summe der Werte für die C-Achsen von Position P1 und Position P2.

△

HINWEIS

In den oben gezeigten Beispielen wird die relative Konvertierung aus Gründen der Anschaulichkeit im zweidimensionalen Raum erläutert. Beim Roboter findet der Vorgang im dreidimensionalen Raum statt. Zusätzlich hängt die Lage des Werkzeugkoordinatensystems dabei von der Stellung des Roboters ab.

4.9 Angehängte Anweisung

Bei Interpolationsbefehlen ist es möglich, eine Verknüpfung an die Anweisung anzuhängen. Durch Anhängen einer Verknüpfung können bestimmte Befehle parallel zum Interpolationsbefehl ausgeführt werden.

Erläuterung

Befehl	Beschreibung
Wth	Während einer Interpolationsbewegung wird eine unbedingte, zusätzliche Anweisung ausgeführt.
WthIf	Während einer Interpolationsbewegung wird eine bedingte, zusätzliche Anweisung ausgeführt.

Anweisungsbeispiele

Mov P1 Wth M_Out(20) = 1	Position P1 anfahren und Ausgangsbit 20 auf „1“ setzen
Mov P1 WthIf M_In(20) = 1, Hlt	Stoppt, falls während der Anfahrt von P1 Eingangsbit 20 auf „1“ gesetzt wird
Mov P1 WthIf M_In(19) = 1, Skip	Stoppt, falls während der Anfahrt von P1 Eingangsbit 19 auf „1“ gesetzt wird und springt zum nächsten Programmschritt

Befehl steht in Beziehung zu folgenden Funktionen:

Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Stopp	⇒	Abschn. 4.5.6

4.10 Unterschiede zwischen MELFA-BASIC V und MELFA-BASIC IV

4.10.1 MELFA-BASIC V

Die Programmierung der Roboter, die mit den Steuergeräten CRn-700 verwendet werden, erfolgt in der Roboterprogrammiersprache MELFA-BASIC V. MELFA-BASIC V ist bedienungsfreundlicher als MELFA-BASIC IV. Im Folgenden werden die Unterschiede zwischen den beiden Versionen erläutert.

4.10.2 Merkmale der Programmiersprache MELFA-BASIC V

- Die in MELFA-BASIC IV üblichen Zeilennummer werden durch Schrittnummern ersetzt, die automatisch eingefügt werden. Operationen, die eine Angabe der Zeilennummer erfordern entfallen somit, die Effizienz der Programmierung erhöht sich. Der Testbetrieb wird vereinfacht, da keine Fehler durch falsche Zeilennummern mehr auftreten können.
- Befehle und Variablen usw. lassen sich nun in Groß-/Kleinschreibung darstellen, was die Lesbarkeit deutlich erhöht.
- Die Programmiersprache wurde um einige Befehle erweitert und die Funktionen wurden verbessert.

Merkmal	MELFA-BASIC V	MELFA-BASIC IV
Programmname	Ein Programmname darf aus Großbuchstaben und Zahlen und maximal 12 Zeichen bestehen. (Auf der Anzeige des Steuergeräts können jedoch nur bis zu 4 Zeichen dargestellt werden. Es empfiehlt sich daher, nur 4 Zeichen zu verwenden.)	
Zeichen, die verwendet werden können	• Buchstaben (Groß- und Kleinbuchstaben) • Zahlen • Symbole	• Buchstaben (Großbuchstaben) (Nur bei Kommentaren oder Zeichenketten ist die Verwendung von Kleinbuchstaben zulässig.) • Zahlen • Symbole
Schrittnummer (Zeilennummer)	Die Schrittnummer wird automatisch hinzugefügt.	Die Zeilennummer muss bei der Programmierung eingegeben werden.
Länge einer Zeile	Weniger als 240 Zeichen	Weniger als 127 Zeichen
Variablennamen	Weniger als 16 Zeichen Für den Variablennamen können Groß- und Kleinbuchstaben verwendet werden. Auch wenn gleiche Buchstaben in einem Variablennamen in Groß- und Kleinschreibung verwendet werden, wird die Schreibweise der ersten Registrierung beim Lesen beibehalten.	Weniger als 8 Zeichen Alle Buchstaben des Variablennamens werden in Großbuchstaben umgewandelt.
Markennamen	Weniger als 16 Zeichen Für den Markennamen können Groß- und Kleinbuchstaben verwendet werden. Auch wenn gleiche Buchstaben in einem Markennamen in Groß- und Kleinschreibung verwendet werden, wird die Schreibweise der ersten Registrierung beim Lesen beibehalten.	Weniger als 8 Zeichen Alle Buchstaben des Markennamens werden in Großbuchstaben umgewandelt.
Befehlsnamen Funktionen Systemstatusvariablen	Die Darstellung erfolgt in Groß- und Kleinbuchstaben. Auch wenn gleiche Buchstaben in Groß- und Kleinschreibung verwendet werden, wird die Schreibweise der ersten Registrierung beim Lesen beibehalten.	Alle Buchstaben werden in Großbuchstaben umgewandelt.
Festlegung von Sprungzielen (GoTo, GoSub)	Durch Marke festgelegt	Durch Marke oder Zeilennummer festgelegt

Tab. 4-4: *Unterschiede zwischen MELFA-BASIC V und MELFA-BASIC IV*

4.11 Multitasking-Funktion

4.11.1 Beschreibung

Die Multitasking-Funktion ermöglicht die parallele Ausführung mehrerer Programme zur Verkürzung der Taktzeiten. Der Roboter kann neben seiner Bewegung weitere Funktionen ausführen und mit der Peripherie kommunizieren, z. B. um Signale weiterzugeben.

Beim Multitasking wird jedem Programm ein Programmplatz (Slot/Task) zugeordnet. Die Definition der Programmplätze erfolgt über die Zuweisung von Programmnamen, Format, Startbedingungen und Priorität eines Programms in den Programmplatzparametern SLT1 bis SLT32.

Die Ausführung des Multitaskings kann über das Steuergerät, einen speziellen Eingang oder über einen auf das Multitasking bezogenen Befehl erfolgen. Insgesamt ist die parallele Ausführung von 32 Programmen möglich. (In der Werkseinstellung ist die Anzahl der Programmplätze auf 8 gesetzt.)

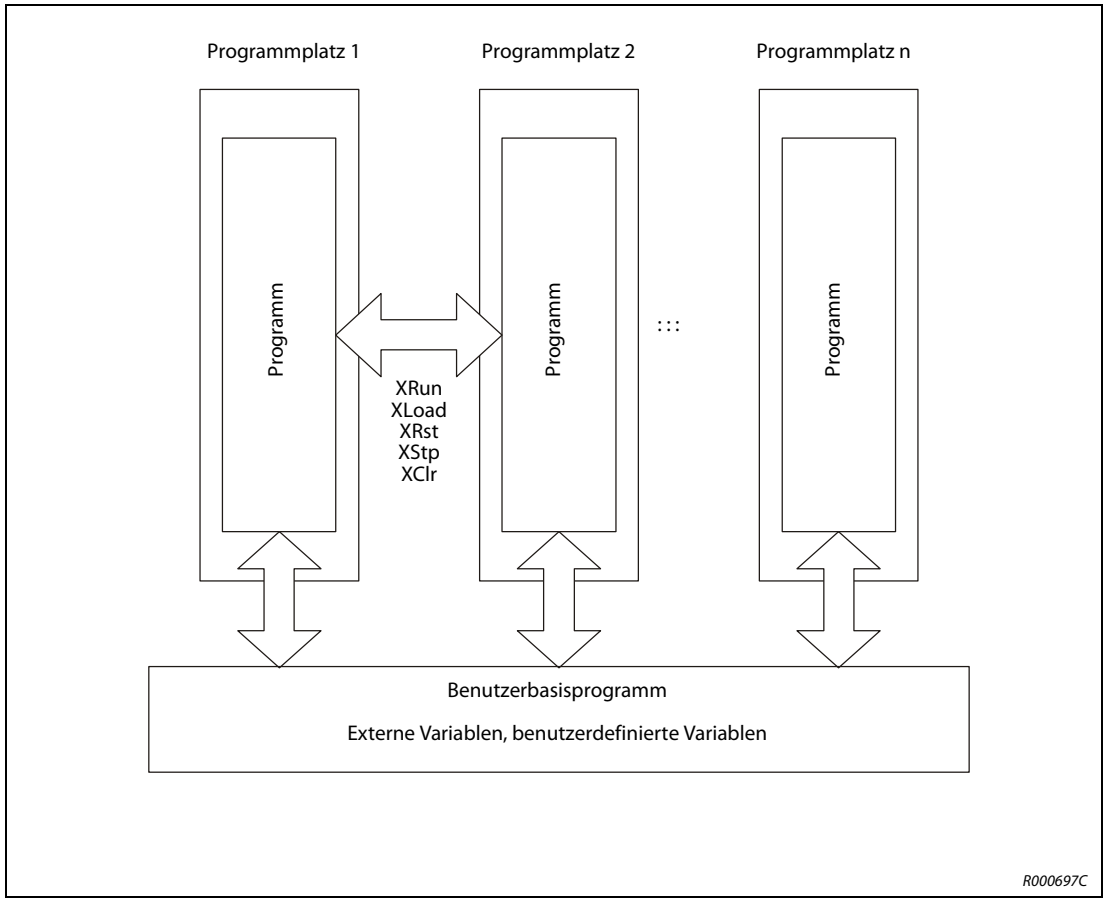


Abb. 4-20: *Multitasking*

HINWEIS

Bei Ausführung eines Programmes wird das Programm einem Programmplatz (Slot/Task) zugeordnet und gestartet. Bei Aufruf eines Programmes über das Bedienfeld des Steuergerätes wird das Programm vom Steuergerät automatisch dem Programmplatz 1 zugeordnet.

4.11.2 Ausführung eines Multitaskings

Zur Ausführung eines Multitaskings stehen drei Möglichkeiten zur Verfügung:

- Ausführung aus einem Programm
  Bei dieser Methode wird die parallele Ausführung von Programmen aus einer nicht festgelegten Position heraus über einen MELFA-BASIC-V-Befehl gestartet. Dabei können die Programme, die parallel ausgeführt oder die Programme, die bei einer parallelen Verarbeitung gestoppt werden sollen, gewählt werden. Diese Methode ist sinnvoll, wenn die Auswahl der Programme aus dem Programmfluss heraus erfolgen soll. Bei dieser Methode werden die Befehle XLoad, XRun, XStp und XRst verwendet. Eine detaillierte Beschreibung der Befehle finden Sie in Abschn. 6.3.

- Ausführung über das Steuergerät oder ein externes Ein-/Ausgangssignal
  Bei dieser Methode wird die parallele Ausführung von Programmen im Startbetrieb, als durchgehender paralleler Betrieb oder als paralleler Betrieb bei Auftreten einer Fehlermeldung in Abhängigkeit der Programmplatzparameter gestartet. Die Programmplatzparameter (SLTn) müssen vor der Ausführung gesetzt werden.
  Diese Methode ist vom Programmfluss unabhängig und für eine gleichzeitige Ausführung mit voreingestelltem Format oder eine sequentielle Ausführung sinnvoll.

- Automatische Ausführung bei Einschalten der Versorgungsspannung
  Bei dieser Methode wird die Ausführung direkt nach dem Booten des Steuergerätes gestartet. Ist im Programmplatzparameter die Startbedingung „ALWAYS“ eingestellt, startet das Programm nach dem Booten automatisch im kontinuierlichen Betrieb. Dadurch werden Probleme beim Starten von Programmen zur Überwachung von Ein- und Ausgangssignalen über Programmplätze durch die SPS verhindert.
  Zusätzlich kann die Ausführung eines Programms aus einem anderen Programm zur kontinuierlichen Steuerung von Roboterbewegungen heraus erfolgen. Setzen Sie zur Freigabe von X-Befehlen, wie XRun oder XLoad, des Servo-Befehls oder des Reset-Befehls den Parameter ALWENA auf „1“.

4.11.3 Betriebszustand eines Programmplatzes

Der Betriebszustand eines Programmplatzes ist von den ausgeführten Operationen und Befehlen abhängig. Jeder Zustand kann über eine Roboterstatusvariable oder ein externes Ausgangssignal angezeigt werden.

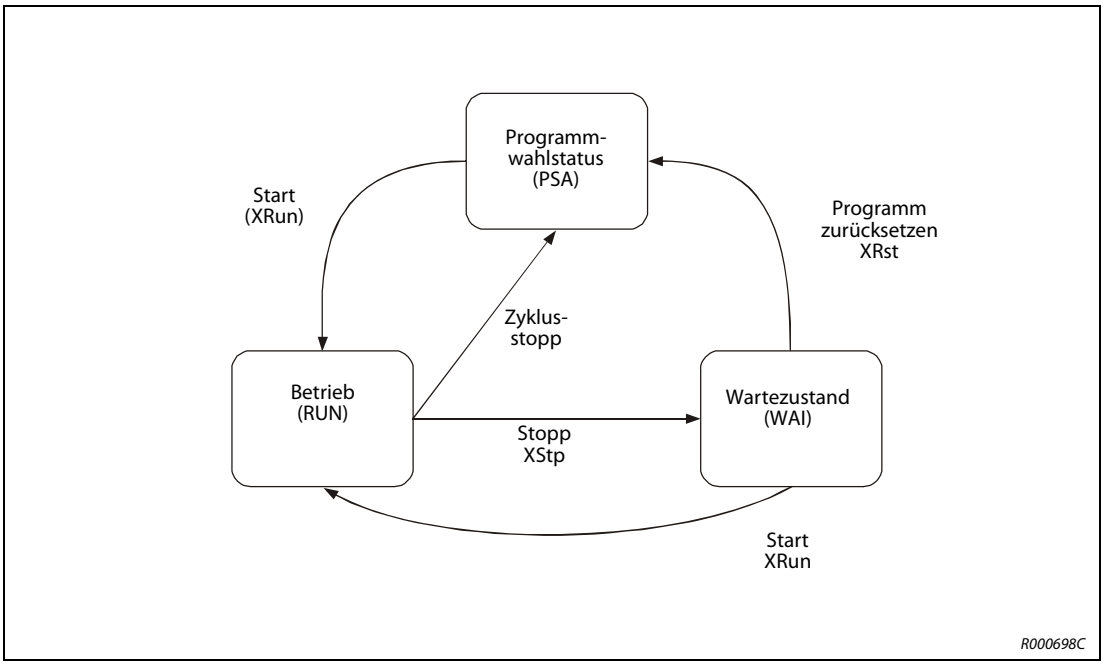


Abb. 4-21: *Betriebszustand eines Programmplatzes*

Programmplatzparameter

Mit Hilfe der Programmplatzparameter SLT1 bis SLT32 können der Programmname, das Format, die Startbedingung und die Priorität für 32 Programmplätze festgelegt werden. (In der Werkseinstellung ist die Anzahl der Programmplätze auf 8 gesetzt.) Eine detaillierte Beschreibung der Parameter finden Sie in Kap. 9.

Die Festlegung erfolgt in der Reihenfolge:
Parametername = 1. Programmname, 2. Ausführungsformat, 3. Startbedingung, 4. Priorität

Programmplatz-parameter	Werks-einstellung	Einstellung	Funktion
Programmname	SLT1: über Steuergerät gewähltes Programm SLT2 bis 32: über Parameter festgelegtes Programm	Einstellung eines registrierten Programms	Über den Parameter kann ein registriertes Programm zur Ausführung im Multitasking festgelegt werden. Ist die Ausführung der Programme von unterschiedlichen Bedingungen abhängig, kann eine Steuerung über die Befehle XLoad und XRun aus einem anderen Programm heraus erfolgen. Das auf dem Steuergerät angewählte Programm ist automatisch SLT1 zugeordnet.
Ausführungs-format	REP	REP: kontinuierlicher Betrieb	Das Programm springt nach der letzten Zeile oder der End-Anweisung an den Anfang zurück und wird erneut ausgeführt.
		CYC: zyklischer Betrieb	Bei Ausführung der End-Anweisung wird das Programm beendet und die Auswahl des Programmes zurückgesetzt. Soll das Programm weiterhin angewählt bleiben, muss der Parameter SLOTON entsprechend eingestellt werden. Detaillierte Hinweise zur Einstellung des Parameters finden Sie in Kap. 9.
Startbedingung	START	START: Startanforderung Die Programmausführung wird über die START-Taste des Steuergerätes oder ein externes Ein-/Ausgangssignal gestartet.	Die Einstellung „START" ist die übliche Einstellung zum Starten einer Programmausführung. ①
		ALWAYS: Ständig Die Programmausführung wird automatisch nach dem Booten des Steuergerätes gestartet.	Die Einstellung „ALWAYS" wird zur kontinuierlichen Ausführung von Programmen verwendet. Bei einem kontinuierlichen Betrieb über die Einstellung „ALWAYS" können keine Befehle, die die Roboterbewegung steuern, z. B. der Mov-Befehl, ausgeführt werden. Ein Programm, das mit der Einstellung „ALWAYS" ausgeführt wird, kann über den Befehl XStp gestoppt werden. Ein Stoppen des Programms über das Bedienfeld des Steuergerätes, ein externes Eingangssignal oder über den NOT-HALT-Schalter ist nicht möglich.
		ERROR: Fehler Die Programmausführung wird bei Auftreten eines Fehlers gestartet.	Die Einstellung „ERROR" wird gewählt, wenn die Programmausführung nach Auftreten eines Fehlers erfolgen soll. Es können keine Befehle, die die Roboterbewegung steuern, z. B. der Mov-Befehl, ausgeführt werden. Das Ausführungsformat (REP/CYC) wird unabhängig von der Einstellung auf einen Zyklus (CYC) gesetzt.

Tab. 4-5: *Programmplatzparameter (1)*

Programmplatz-parameter	Werks-einstellung	Einstellung	Funktion
Priorität (Anzahl der auszuführenden Zeilen)	1	1 bis 31: Die Einstellung legt die Anzahl der auszuführenden Zeilen für einen Durchgang fest.	Je größer der Einstellwert, desto mehr Zeilen des Programms werden bei einem Durchgang ausgeführt. Ist die Priorität z. B. für SLT1 auf „10“, für SLT 2 auf „5“ und für SLT 3 auf „1“ gesetzt, werden nach Abarbeitung von 10 Zeilen des Programms in SLT1 5 Zeilen des Programms in SLT2 und 1 Zeile des Programms in SLT3 ausgeführt. Nach einem Durchgang beginnt dieser Zyklus von vorne.

Tab. 4-5: *Programmplatzparameter (2)*

① Beim Start über das Steuergerät oder über das spezielle Eingangssignal START werden die Programme aller Programmplätze gestartet, für die ein Starten über Startanforderung definiert wurde. Ein unabhängiger Programmstart erfolgt über die Startsignale S1START bis S32START. In diesem Fall ist die Schrittnummer dem Ein-/Ausgangsparameter zugewiesen. Eine detaillierte Beschreibung der speziellen Parameter für die Ein-/Ausgänge finden Sie in Abschn. 10.3.

Beispiel ▽ Dieses Beispiel zeigt die Parametereinstellungen zur Festlegung der folgenden Bedingungen für Programmplatz 2:

Programmname: 5
Ausführungsformat: kontinuierlicher Betrieb
Startbedingung : ständig
Priorität: 10

Einstellung: SLT 2 = 5, REP, ALWAYS, 10

△

4.11.4 Erstellung eines Multitasking-Programms

Anzahl der Programmplätze und Verarbeitungszeit

Bei der Ausführung eines Multitaskings scheinen alle gestarteten Programme gleichzeitig ausgeführt zu werden. In Wirklichkeit erfolgt jedoch nur jeweils die Verarbeitung einer einzelnen Zeile und die Programmsteuerung springt von einem Programm in das nächste. Die Anzahl der in einem Programm nacheinander abzuarbeitenden Zeilen kann über die Programmplatzparameter SLT festgelegt werden (siehe auch Kap. 9). Je mehr Programme also ausgeführt werden, desto größer wird die Verarbeitungszeit. Die Programmzahl beim Multitasking ist daher aus Gründen der Effizienz auf ein Minimum zu begrenzen. Programme für andere Anwendungen, z. B. zur Ausführung von Roboterbewegungsbefehlen (Mov oder Mvs) können jedoch zu jeder Zeit ausgeführt werden.

Anzahl der Programme, die parallel ausgeführt werden sollen

Die Einstellung der Anzahl der Programme, die parallel ausgeführt werden sollen, erfolgt über den Parameter TASKMAX (Werkseinstellung: 8). Sollen mehr als 8 Programme parallel ausgeführt werden, muss der Parameterwert geändert werden. Es ist jedoch sinnvoll die Anzahl der parallel auszuführenden Programme an die realen Bedingungen anzupassen; sind z. B. nur 2 Programme in der parallelen Ausführung, so sollte dieser Parameter auf den Wert „2“ gestellt werden, um eine Verringerung der Zykluszeit zu bewirken.

Datenaustausch zwischen Programmen über externe Variablen

Über externen Variablen wie M_00 oder P_00 und benutzerdefinierte Variablen kann ein Datenaustausch zwischen den im Multitasking betriebenen Programmen stattfinden. Eine detaillierte Beschreibung der externen Variablen finden Sie in Abschn. 5.1.10.

Beispiel▽ Dieses Beispiel zeigt eine Steuerung des EIN/AUS-Zustandes des Eingangsbits 8 über Programmplatz 2. Der EIN-Zustand wird über die externe Variable M_00 an Programmplatz 1 übertragen.

Programmplatz 1 (Slot 1)

```
1  M_00 = 0                  'Setzt die Variable M_00 auf „0“
2  *L                        'Marke L festgelegt
3  If M_00 = 0 THEN *L       'Wartestatus, bis M_00 ungleich „0“
4  M_00 = 0                  'Setzt die Variable M_00 auf „0“
5  Mov P1                    'Fortsetzung des normalen Betriebs
6  Mov P2
      :
10 GoTo *L                   'Wiederhole ab Schritt 2
```

Programmplatz 2 (Slot 2)

```
1  If M_IN(8) <> 1Then *A1   'Sprung zu Schritt 3, falls Eingangsbit 8 nicht
                             EIN ist
2  M_00 = 1                  'Setzt die Variable M_00 auf „1“
3  *A1                       'Marke A1 festgelegt
4  Mov P1                    'Fortsetzung des normalen Betriebs
      :
```

△

Überwachung des Programmstatus über Roboterstatusvariablen

Über die Roboterstatusvariablen (M_Run, M_Wai und M_Err) kann der Programmstatus einer im Multitasking betriebenen Anwendung von jedem Programmplatz aus überwacht werden.

Beispiel▽

```
M1 = M_Run(2)          'Schreibt den Programmstatus des Programmplatzes 2 in M1
```

Eine detaillierte Beschreibung der Roboterstatusvariablen finden Sie auf Seite 5-18.

△

Externe Signalein- und -ausgänge

Die externen Signalein- und -ausgänge können von jedem Programmplatz verwendet werden.

Programmplatz 1

Das Hauptsteuerprogramm, in dem die Bewegungsbefehle des Roboters (Mov-Befehle usw.) festgelegt sind, wird in der Regel Programmplatz 1 zugeordnet. Eine andere Zuordnung muss über die Befehle GetM und RelM erfolgen. Eine detaillierte Beschreibung der Befehle finden Sie in Abschn. 6.3.

Initialisierung kontinuierlich ausgeführter Programme

Programme, deren Startbedingung über die Programmplatzparameter auf „ALWAYS“ gesetzt sind, werden kontinuierlich ausgeführt, wenn das Ausführungsformat auf „REP“ eingestellt ist. Die Programme sollten so aufgebaut sein, dass die Initialisierung nur einmal im Programm erfolgt, indem z.B. eine Verzweigung in Abhängigkeit eines Initialisierung-beendet-Merkers ausgeführt wird. (Das gilt nicht für Programme, deren Ausführungsformat auf „CYC“ (1 Zyklus) gesetzt ist, da das Programm nur einmal ausgeführt wird.)

HINWEIS

In der Grundeinstellung wird der Mechanismus 1 (Roboter bei Standardsystem) dem Programmplatz 1 zugeordnet. Bewegungsbefehle können somit ohne Zuordnung über den GetM-Befehl im Programmplatz 1 definiert werden. Sollen Bewegungsbefehle in einer Anwendung ausgeführt werden, so ist zuerst die Zuordnung von Programmplatz und Mechanismus über den Befehl RelM aufzuheben und anschließend die neu gewünschte Zuordnung über den Befehl GetM zu definieren.

4.11.5 Anwendung des Multitaskings

Multitasking starten

Bei Ausführung des Startvorgangs über das Bedienfeld der Steuereinheit oder den speziellen Eingang START starten alle Anwendungen gleichzeitig, für die in den Programmplatzparametern die Startbedingung „Startanforderung“ festgelegt wurde. Ein separates Starten der Programme ist über die Starteingänge S1START bis S32START möglich. In diesem Fall ist die Schrittnummer dem Ein-/Ausgangsparameter zugewiesen. Eine detaillierte Beschreibung der speziellen Parameter für die Ein-/Ausgänge finden Sie in Abschn. 10.3.

Anzeige des Betriebszustandes

Die LEDs der Taster START und STOP auf dem Bedienfeld des Steuergeräts und die speziellen Ein-/Ausgangssignale START und STOP zeigen den Betriebszustand aller Programme an, deren Startbedingung in den Programmplatzparametern SLTn auf „START“ gesetzt ist. Bei Ausführung eines Programmes leuchtet die LED des START-Tasters und das Ausgangssignal START wird eingeschaltet. Bei einem Programmstopp leuchtet die LED des STOP-Tasters und das Ausgangssignal STOP wird eingeschaltet.
Die speziellen Ausgangssignale S1START bis S32START und S1STOP bis S32STOP zeigen den Betriebszustand jedes einzelnen Programmplatzes an. Zur unabhängigen Anzeige der Betriebszustände muss die entsprechende Zeilennummer dem Ein-/Ausgangsparameter zugewiesen werden. Eine detaillierte Beschreibung der speziellen Parameter für die Ein-/Ausgänge finden Sie in Abschn. 10.3.
Der Betriebszustand eines Programms mit der Startbedingung auf „ALWAYS“ oder „ERROR“ wird nicht über die LEDs der Taster START und STOP angezeigt. Der Betriebszustand kontinuierlich ausgeführter Programme kann über die optionale Programmier-Software des Roboters angezeigt werden.

4.11.6 Beispiel zur Anwendung der Multitasking-Funktion

Detaillierte Beschreibung des Arbeitsablaufs

Der Arbeitsablauf wird in zwei Programme aufgeteilt:

- Programm mit Bewegungsbefehlen
  Dem Programm, das die Bewegungsbefehle enthält, ist der Programmplatz 1 zugewiesen.
- Programm zum Einlesen von Positionsdaten
  Dem Programm zum Einlesen der Positionsdaten ist der Programmplatz 2 zugewiesen. Wird während der Roboterbewegung ein Startsignal an einen Sensor ausgegeben, erfolgt über das Programm zum Einlesen von Positionsdaten eine Datenabfrage des Personalcomputers. Der Personalcomputer überträgt die Positionsdaten über Programmplatz 2 zum Roboter.

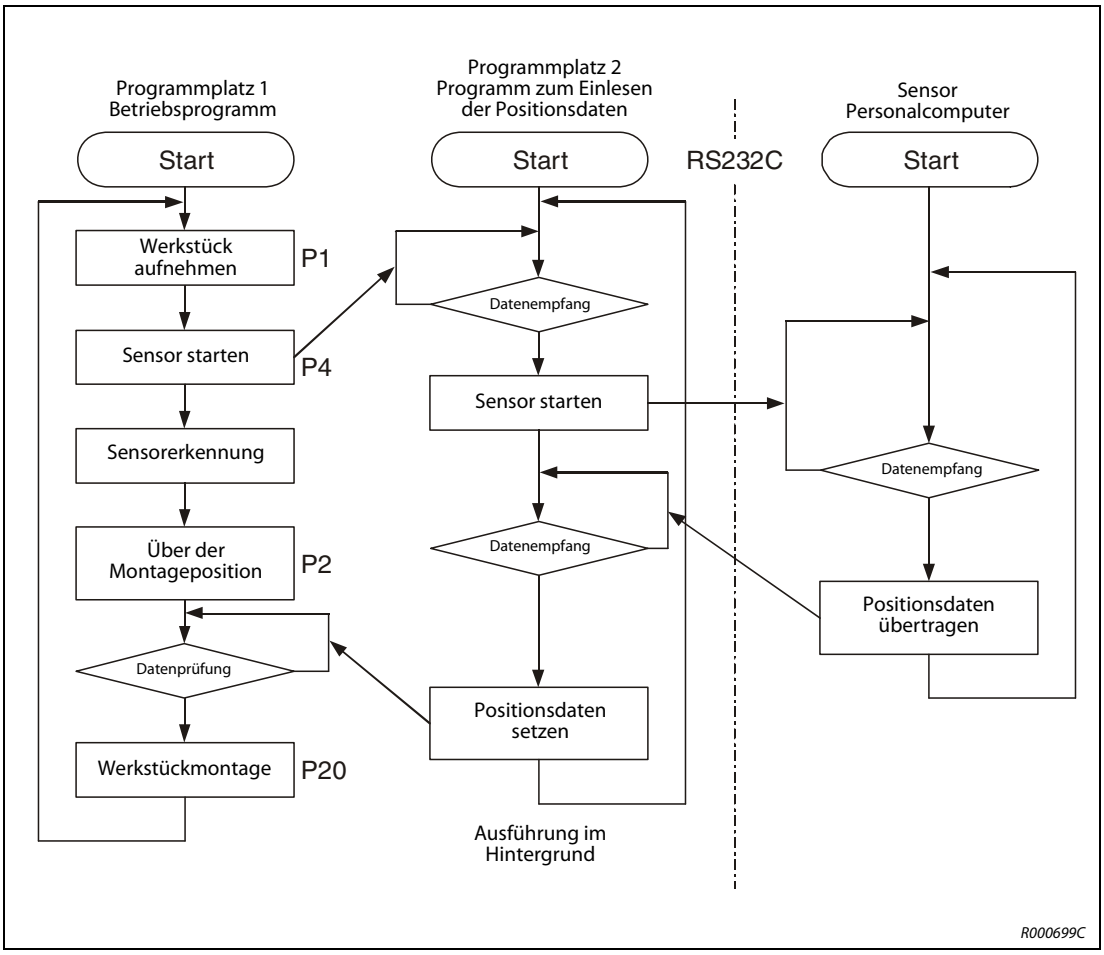


Abb. 4-22: *Flussdiagramm*

Positionen

P1: Aufnahme des Werkstücks (Wartezeit Vakuumgreifer Dly 0.05)
P2: Ablage des Werkstücks (Wartezeit Dly 0.05)
P3: Position vor der Überwachung (Position ohne Stopp durchlaufen Cnt)
P4: Position nach der Überwachung (Position ohne Stopp durchlaufen Cnt)
P_01: Kompensationsdaten der Überwachung
P20: Relative Konvertierung der Kompensationsdaten der Überwachung P_01 und P2

Bewegungsablauf

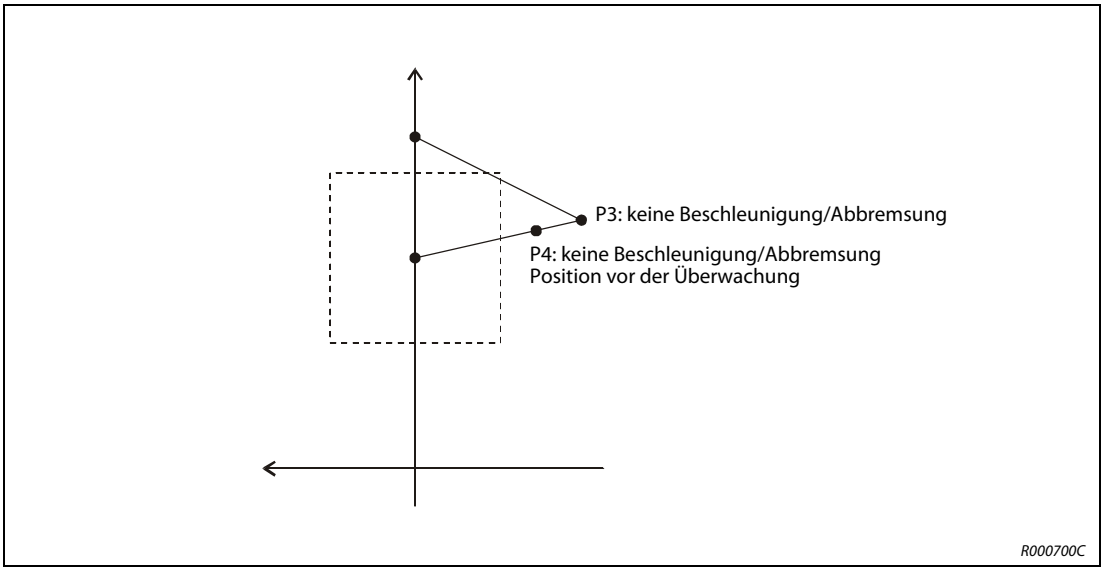


Abb. 4-23: *Verfahrbewegung und Positionen*

Programm mit Bewegungsbefehlen (Programmplatz 1)

```
1  Cnt 1                          'Überschleiffunktion freigeben
2  Mov P2, 10                     'Position 10 mm über P2 anfahren
3  Mov P1, 10                     'Position 10 mm über P1 anfahren
4  Mov P1                         'Aufnahmeposition P1 anfahren
5  M_Out(10) = 0                  'Werkstück aufnehmen
6  Dly 0.05                       'Wartezeit 0,05 s
7  Mov P1, 10                     'Position 10 mm über P1 anfahren
8  Mov P3                         'Position P3 anfahren
9  Spd 500                        'Geschwindigkeit auf 500 mm/s setzen
10 Mvs P4                         'Mit Überfahren von P4 Überwachung starten
11 M_02# = 0                      'Hintergrundprozess zum Einlesen der Daten
                                  mit Sperrvariablen (M_01 = 1/M_02 = 0) starten
18 M_01# = 1                      'Einlesen der Daten im Hintergrund starten
19 Mvs P2, 10                     'Position 10 mm über P2 anfahren
20 *L                             'Sprungmarke L festgelegt
21 If M_02# = 0 Then GoTo *L      'Warten, bis Sperrvariable M_02 gleich 1
22 P20 = P2 * P_01                'Multipliziere P_01 mit P2
23 Mov P20, 10                    'Position 10 mm über P20 anfahren
24 Mov P20                        'Ablageposition P20 anfahren
25 M_Out(10) = 1                  'Werkstück ablegen
26 Dly 0.05                       'Wartezeit 0,05 s
27 Mov P20, 10                    'Position 10 mm über P20 anfahren
28 Cnt 0                          'Überschleiffunktion sperren
29 End                            'Zyklusende
```

Programm zum Einlesen von Positionsdaten (Programmplatz 2)

```
1  *R                          'Sprungmarke R festgelegt
2  If M_01# = 0 Then GoTo *R   'Warten, bis Sperrvariable M_01 gleich 1
3  Open "COM1:" AS #1          'Öffnet die RS232C-Kommunikationsschnittstelle
4  Dly M_03#                   'Hypothetische Wartezeit (0,05 s)
5  Print #1, "SENS"            'Zeichenkette „SENS“ über RS232C-Schnittstelle
                               ausgeben (Anzeige)
6  Input #1, M1, M2, M3        'Wartet auf Einlesen der Kompensationsdaten
                               (relative Daten)
7  P_01.X = M1                 'Überschreiben der ΔX-Koordinate
8  P_01.Y = M2                 'Überschreiben der ΔY-Koordinate
9  P_01.Z = 0.0                '
10 P_01.A = 0.0                '
11 P_01.B = 0.0                '
12 P_01.C = RAD(M3)            'Überschreiben der ΔC-Koordinate
13 Close                       'Kommunikationsschnittstelle schließen
14 M_01# = 0                   'Setzen der Sperrvariablen M_01 = 0
15 M_02# = 1                   'Setzen der Sperrvariablen M_02 = 1
16 End                         'Programm beenden
```

Einstellung der Programmplatzparameter

SLT 1 = 1, REP, START, 1
SLT 2 = 2, REP, START, 2

Zur Aktivierung der Programmplatzparameter muss die Versorgungsspannung aus und wieder eingeschaltet werden.

Start

Der Start von Programm 1 und 2 erfolgt über das Bedienfeld des Steuergerätes.

4.11.7 Programmspeicherkapazität

Der Programmspeicher ist in drei Bereiche eingeteilt:

- Speicher für die Programmsicherung
  In diesem Bereich werden Programme gespeichert. Im Regelfall können hier insgesamt 920 Kilobytes Programmcode gespeichert werden. Ein Zusatzspeicher ermöglicht eine Erweiterung des Bereichs auf 2 Megabytes.
- Speicher für die Programmeditierung
  Dieser Bereich wird für die Editierung und die schrittweise Ausführung von Programmen verwendet. Die Speichergröße beträgt 380 Kilobytes und entspricht der maximalen Größe eines Programms. Die Größe des Speichers für die Programmeditierung kann auch durch einen Zusatzspeicher nicht erhöht werden.
- Speicher für die Programmausführung
  Dieser Speicherbereich wird bei der automatische Ausführung von Programmen verwendet. Die Speichergröße beträgt 400 Kilobytes. Die maximal mögliche Belegung des Speichers durch ein Benutzerbasisprogramm, durch Multitasking-Aufgaben oder durch die Anweisungen XRun und CallP beträgt 400 Kilobytes. Die Größe des Speichers für die Programmausführung kann auch durch einen Zusatzspeicher nicht erhöht werden.

Speicherbereich	Kapazität	
	Standardspeicher	Mit Erweiterungsspeicher
Speicher für die Programmsicherung	920 KB	2 MB
Speicher für die Programmeditierung	380 KB	
Speicher für die Programmausführung	400 KB	

Tab. 4-6: *Kapazitäten der einzelnen Speicherbereiche*

HINWEIS

Die Größe jedes Programms kann über die Teaching Box und das Programmverwaltungsmenü der Roboter-Programmier-Software RT ToolBox2 überprüft werden.

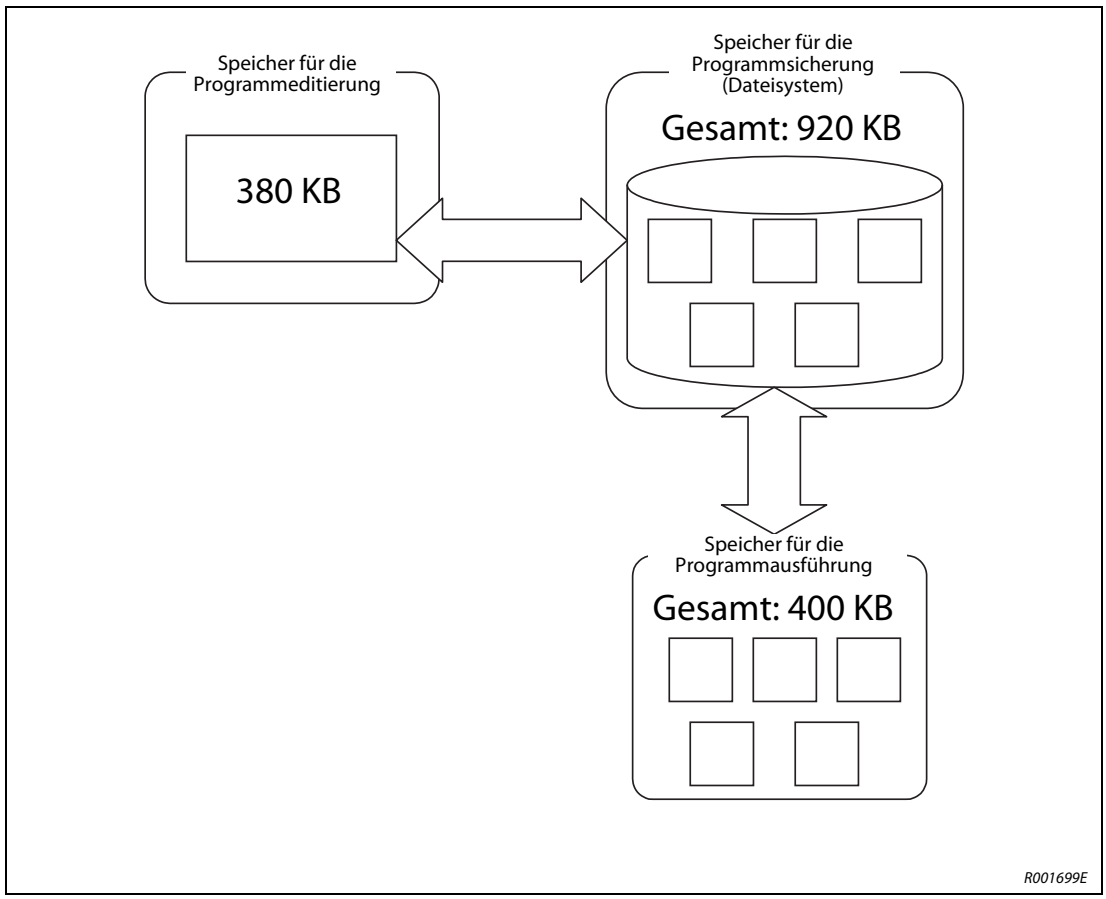


Abb. 4-24: *Kapazität des Standardspeichers*

5 MELFA-BASIC V

Die nachfolgenden Abschnitte enthalten eine Auflistung aller in MELFA-BASIC V verwendeten Datentypen und deren Anwendungsmöglichkeiten.

5.1 Begriffserklärung

5.1.1 Anweisung

Eine Anweisung ist die kleinste Einheit eines Programms. Sie besteht aus einem Befehl und einem Befehlsparameter.

Beispiel▽ Mov P1 = Anweisung
Mov = Befehl
P1 = Befehlsparameter

△

5.1.2 Angehängte Anweisung

Bei Interpolationsbefehlen ist es möglich, eine Verknüpfung an die Anweisung anzuhängen. Durch Anhängen einer Verknüpfung können bestimmte Befehle parallel zum Interpolationsbefehl ausgeführt werden. Es darf pro Zeile nur eine Verknüpfung angehängt werden.

Beispiel▽ Folgender Befehl bewirkt, dass die Position P1 mittels Gelenk-Interpolation angefahren und gleichzeitig das Ausgangsbit 17 auf „1“ gesetzt wird:
Mov P1 Wth M_Out(17) = 1

△

Mit Hilfe der Befehle Wth und WthIf können Verknüpfungen an eine Anweisung angehängt werden.

5.1.3 Programmschritt

Ein Programmschritt besteht aus einer Schrittnummer und einer oder zwei Anweisungen, wenn zusätzlich eine Konjunktion angehängt ist.

Ein Programmschritt darf aus maximal 240 Zeichen bestehen. Das Zeilenendzeichen wird dabei nicht mitgezählt.

HINWEIS

In der MELFA-BASIC-Programmiersprache ist es nicht erlaubt, mehrere durch Semikolons getrennte Anweisungen in einem Programmschritt zu verwenden, wie es bei vielen BASIC-Dialekten möglich ist.

5.1.4 Schrittnummern und Marken

Schrittnummern

Für die einwandfreie Funktion eines Programms müssen die Schrittnummern in aufsteigender Reihenfolge angeordnet sein. Beim Abspeichern wird das Programm in dieser Reihenfolge im Speicher abgelegt. Der Wertebereich für die Schrittnummern beträgt 1 bis 32 767.

Eine Ausnahme bildet die direkte Befehlsausführung:

HINWEIS

Bei fehlender Schrittnummer wird eine Anweisung direkt nach Betätigung der Eingabetaste ausgeführt. Die Anweisung wird dabei nicht gespeichert.

Marken

Eine Marke ist ein benutzerdefiniertes Wort, das ein Sprungziel festlegt. Erzeugt wird eine Marke durch Eingabe des Asterisk-Zeichens (*) hinter der Schrittnummer und einer alphanumerischen Zeichenkette. Hierbei wird auch zwischen Groß- und Kleinbuchstaben unterschieden. Beginnt die Zeichenkette mit dem Zeichen „L“, kann als nächstes Zeichen der Unterstrich (_) verwendet werden. Das erste Zeichen muss ein Buchstabe sein. Es werden nur die ersten 8 Zeichen ausgewertet.

Beispiel▽

```
12 *ABLAGE;
17 *LAGE_1
```

△

HINWEIS

Für Markennamen dürfen keine reservierten Wörter (z. B. Dly, HOpen usw.), keine Namen, die mit einem Symbol oder einer Zahl beginnen, keine Namen, die schon für eine Variable oder eine Funktion vergeben wurden und kein Unterstrich als zweites Zeichen verwendet werden.

5.1.5 Zeichentypen

Die in MELFA-BASIC V verwendbaren Zeichentypen sind in Tab. 5-1 aufgeführt. Detaillierte Beschreibungen zu Programmnamen finden Sie in Abschn. 4.2.1, zu Variablennamen in Abschn. 5.1.9 und zu Markennamen in Abschn. 5.1.4.

Kategorie	Verwendbare Zeichen in MELFA-BASIC V	Programm-namen	Variablen-namen	Marken-namen
Buchstaben	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	✔	✔	✔
	a b c d e f g h i j k l m n o p q r s t u v w x y z	—	✔	✔
Zahlen	0 1 2 3 4 5 6 7 8 9	✔	①	②
Symbole	" ' & ( ) * + – . , / : ; = < > ? @ ` [\ ] ^ { } ~ |	—	—	—
	! # $ %	—		—
	_ (Unterstrich)	—	③	④
Leerstellen	Leerzeichen	—	—	—

Tab. 5-1: *In MELFA-BASIC V verwendbare Zeichen*

✔ : verwendbar
— : nicht verwendbar

① Ein Variablenname muss mit einem Buchstaben beginnen. Alle weiteren Zeichen können auch Zahlen sein.

② Ein Markennamen muss mit einem Buchstaben beginnen. Alle weiteren Zeichen können auch Zahlen sein.

③ Der Unterstrich kann in Variablennamen als zweites und für die folgenden Zeichen verwendet werden. Alle Variablen mit einem Unterstrich als zweitem Zeichen sind externe Variablen.

④ Der Unterstrich kann in Markennamen ab dem dritten Zeichen verwendet werden.

5.1.6 Zeichen mit besonderer Bedeutung

Unterstrich (_)

Der Unterstrich wird bei Variablennamen als zweites Zeichen verwendet, wenn diese als programmexterne Variablen benutzt werden.

Beispiele▽

```
P_Curr
M_01
M_Abc
```

△

Apostroph (')

Der Apostroph wird vor einen Kommentar gesetzt. Er hat die gleiche Funktion wie der Rem-Befehl (Kennzeichnung eines Kommentars).

Beispiele▽

```
1 Mov P1   'GET          GET wird als Kommentar deklariert
2          'GET PARTS    Entspricht der Zeile: 2 Rem GET PARTS
```

△

Asterisk (*)

Das Asterisk-Zeichen wird vor alle Sprungmarken gesetzt. Die Sprungmarke darf maximal aus 16 Zeichen bestehen.

Beispiel▽

```
2 *LADEN
```

△

Komma (,)

Das Komma dient bei Angabe mehrerer Parameter oder Suffixe zur Trennung.

Beispiel▽

```
P1 = (100, 150, ...)
```

△

Punkt (.)

Der Punkt dient als Dezimalpunkt und zur Unterteilung der einzelnen Komponenten bei mehrteiligen Daten wie Positions- und Gelenkvariablen.

Beispiel▽

```
M1 = P2. X          Schreibt die X-Koordinate der Positionsvariablen P2 in die Variable M1
```

△

Leerzeichen

In Zeichenketten und Kommentaren wird das Leerzeichen wie jedes andere Zeichen interpretiert. Zwischen einzelnen Daten, nach Schrittnummern und Anweisungen dient es zur Trennung.

Im Eingabeformat bei der detaillierten Befehlsbeschreibung (siehe Abschn. 6.3) wird ein notwendiges Leerzeichen durch das Zeichen „□“ dargestellt.

5.1.7 Datentypen

Datentypen umfassen Werte, Positionsdaten, Gelenkdaten und Zeichen. Bei Zahlen unterscheidet man zwischen reellen und ganzen Zahlen.

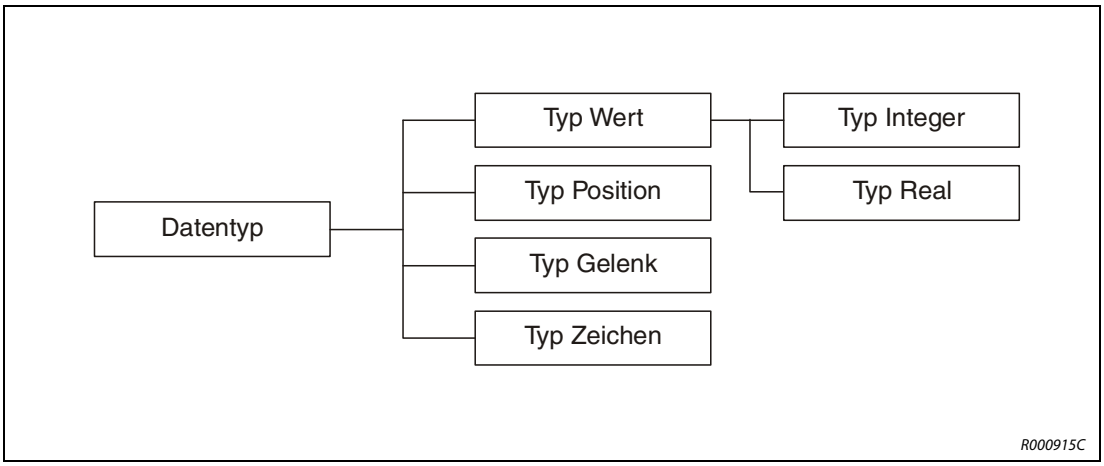


Abb. 5-1: *Datentypen*

5.1.8 Konstanten

Man unterscheidet fünf Arten von Konstanten:

- Numerische Konstanten
- Alphanumerische Konstanten
- Positionskonstanten
- Gelenkkonstanten
- Winkelkonstanten

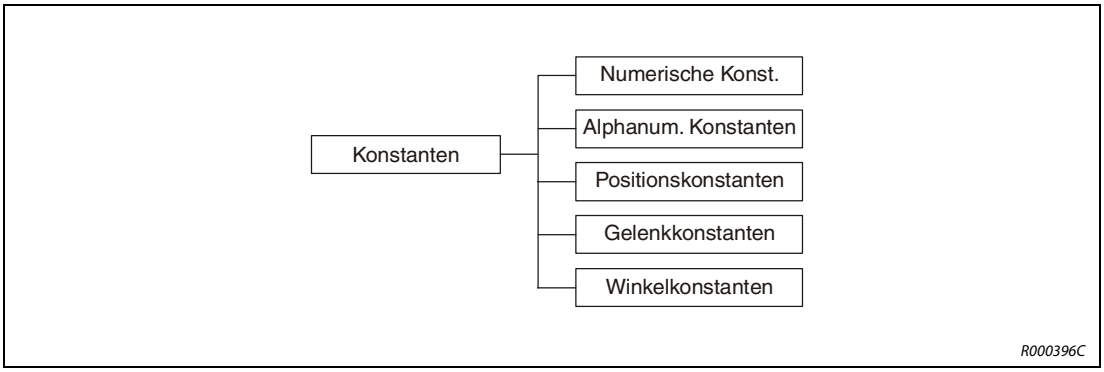


Abb. 5-2: *Konstanten*

Numerische Konstanten

Numerische Konstanten sind wie folgt aufgebaut:

- Dezimalzahlen

Beispiele▽ 1, 1.7, −10,5, +1.2E+5 (Exponentialdarstellung)
Der Wertebereich für Zahlen ist −1.7976931348623157E+308 bis 1.7976931348623157E+308

△

- Hexadezimale Zahlen

Beispiele▽ &H132; &HC011; &H1AC4
Der Wertebereich hexadezimaler Zahlen ist &H0000 bis &HFFFF.

△

- Binäre Zahlen

Beispiele▽ &B010011; &B1101
Der Wertebereich binärer Zahlen ist &B0000000000000000 bis &B1111111111111111.

△

Der Typ einer Konstanten wird durch ein Symbol am Ende der Konstanten definiert:

Beispiele▽ 10% (Typ Integer)
1.0005! (Typ mit einfacher Genauigkeit)
10.000000003# (reelle Zahl mit doppelter Genauigkeit)

△

Zeichenkettenkonstanten

Zeichenketten werden in Anführungszeichen dargestellt (").

Beispiele▽ "ABCDEFGHIJKLMN"; "123"

△

HINWEIS Eine Zeichenkette kann aus bis zu 240 Zeichen bestehen. Dies umfasst auch die Schrittnummer und die Anführungszeichen. Soll die Zeichenkette selbst ein Anführungszeichen enthalten, muss das Zeichen zweimal hintereinander eingegeben werden: Für die Zeichenkette AB"CD, muss "AB""CD" eingegeben werden.

Positionskonstanten

Folgende Abbildung zeigt die Syntax der Positionskonstanten:

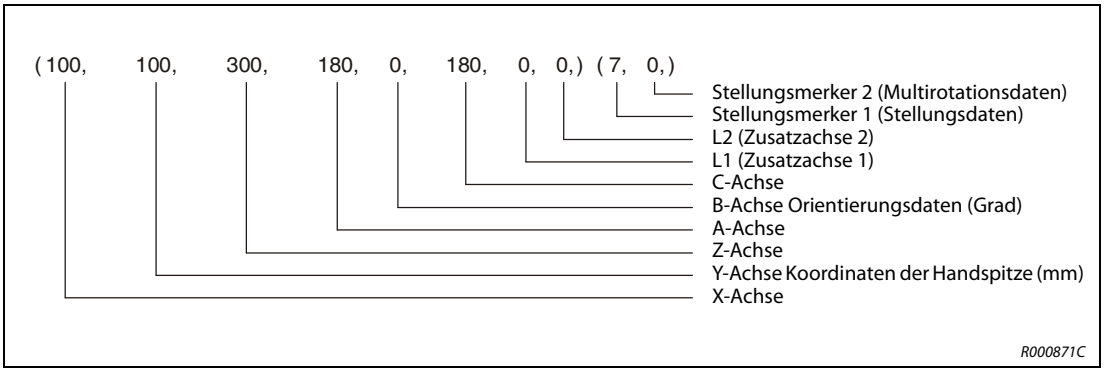


Abb. 5-3: *Positionskonstanten*

Beispiele▽ P1 = (300,100,400,180,0,180,0,0)(7,0)
P2 = (0,0,5,0,0,0)(0,0) [Beispiel ohne Angabe der Zusatzachsendaten]
P3 = (100,200,300,0,0,90)(4,0) [Beispiel für 4-achsigen SCARA-Roboter]

△

Die Koordinaten, Stellungsdaten und die zusätzlichen Achsendaten haben folgende Struktur und Bedeutung:

Struktur: X, Y, Z, A, B, C, L1, L2

- X, Y, Z sind die Daten der Koordinaten. Sie geben die Position der Handspitze (TCP = Tool Center Point) des Roboters im kartesischen Koordinatensystem wieder. Sie werden in mm angegeben.
- A, B, C sind Orientierungsdaten der Roboterhand. Sie geben die Orientierung der Hand im Raum wieder. Ihre Einheit ist Grad. Die Teaching Box, die Roboter-Programmier-Software oder MELFA-Works zeigen die Einheit Grad an. Programminterne Substitutionen und Berechnungen werden jedoch in Radiant ausgeführt.
- L1, L2 sind Zusatzachsendaten. Sie geben die Koordinaten für die zusätzlichen Achsen 1 und 2 an. Ihre Einheit ist mm oder Grad.

Bedeutung der Stellungsdaten (fehlen die Stellungsdaten, werden die Standardeinstellungen (7,0) verwendet (modellabhängig)):

Struktur: FL1, FL2

- FL1: Stellungsmerker

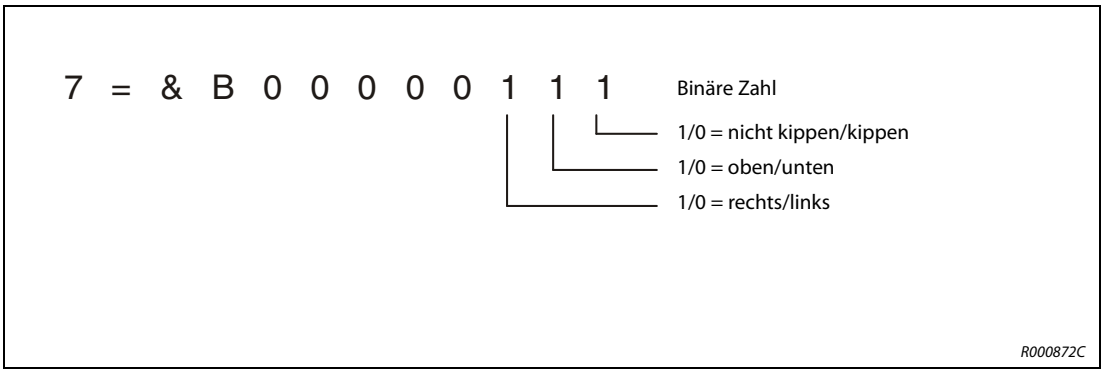


Abb. 5-4: *Bedeutung der Stellungsmerker*

Beispiele▽

```
P1 = (X,Y,Z,A,B,C)(FL1,FL2)
P1 = (X,Y,Z,A,B,C,L1,L2)(FL1,FL2)
```

△

- FL2: Multirotationsinformation

  Standardeinstellung = 0
  Beinhaltet Informationen über die Drehwinkel für jede Gelenkachse in den Positionen (X, Y, Z) mit den Stellungsdaten (A, B, C) als X-, Y- und Z-Koordinaten.
  (Der Einstellbereich ist 0 bis +429496725. Informationen für 8 Achsen werden mit 4 Bits für eine Achse dargestellt. Für den PC existieren 2 Anzeigemöglichkeiten: Anzahl der Rotationen (–8 bis 7) für jede Achse in dezimaler oder in hexadezimaler Form.

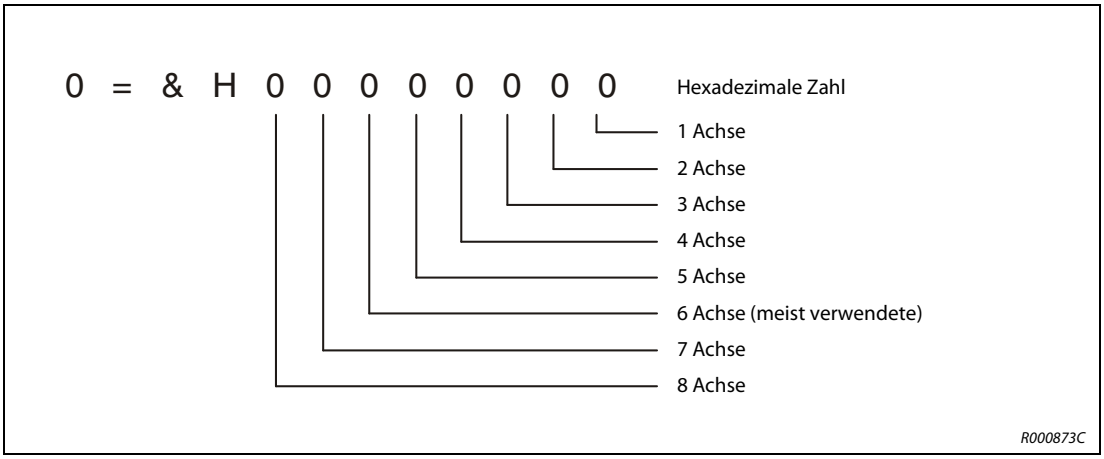


Abb. 5-5: *Multirotationsdaten in hexadezimaler Form*

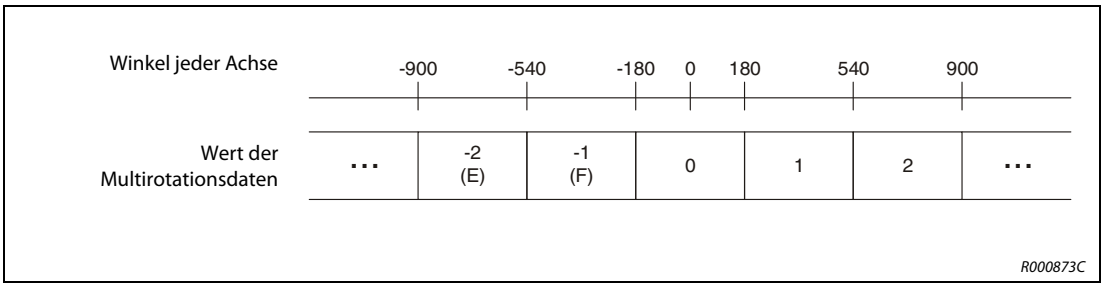


Abb. 5-6: *Wert der Multirotationsdaten*

HINWEISE

Der Wert für das Handgelenk entspricht im XYZ-Koordinatesystem (J6-Achse bei vertikalen Knickarmrobotern) nach einer Drehung von 360° wieder dem Ursprungswert. Aus diesem Grund erfasst FL2 die Anzahl der Drehungen.

Es ist nicht erforderlich, die Koordinaten und Stellungsdaten für sämtliche 8 Achsen anzugegeben. Bei unvollständigen Angaben werden die folgenden Achsendaten als undefiniert verarbeitet. Geben Sie die Daten für einen 4-achsigen Roboter (Achsenkonfiguration: X, Y, Z, C) wie folgt an: (X, Y, Z, , , C) oder (X, Y, Z, 0, 0, C).

Eine Positionskonstante darf keine Variable als Element enthalten.

Gelenkkonstanten

Folgende Abbildung zeigt die Syntax der Gelenkkonstanten:

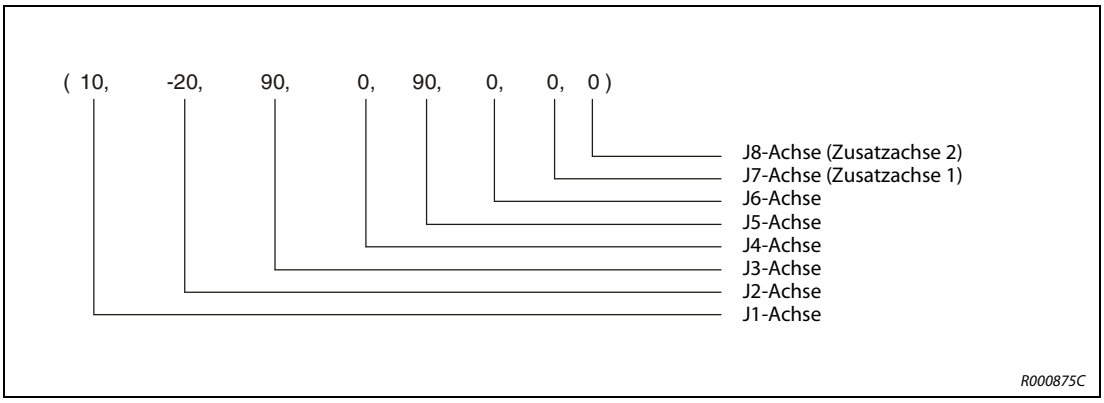


Abb. 5-7: *Gelenkkonstanten*

Beispiele▽

6-achsiger Roboter	J1 = (0,10,80,10,90,0)
6-achsiger Roboter mit Zusatzachsen	J1 = (0,10,80,10,90,0,10,10)
5-achsiger Roboter	J1 = (0,10,80,0,90,0)
5-achsiger Roboter mit Zusatzachsen	J1 = (0,10,80,0,90,0,10,10)
4-achsiger Roboter	J1 = (0,20,90,0)
4-achsiger Roboter mit Zusatzachsen	J1 = (10,20,90,0, , ,10,10)

△

Struktur und Bedeutung der Achsendaten:

Struktur: J1, J2, J3, J4, J5, J6, J7, J8

- J1 bis J6: Achsendaten (in mm oder Grad)
- J7 und J8: Daten der optionalen Zusatzachsen (je nach Einstellung des Parameters AXUN in mm oder Grad)
  Bei SCARA-Robotern mit direkt angetriebener J3-Achse ist die Einheit mm.

HINWEIS Eine Gelenkkonstante darf keine Variable als Element enthalten.

Winkelbetrag

Die Angabe des Winkelbetrages erfolgt in Grad (nicht in Radiant). Bei einer Schreibweise 100DEG wird der Wert als Winkel interpretiert und kann so in trigonometrischen Funktionen verarbeitet werden.

Beispiel▽ Der Sinus eines 90°-Winkels wird folgendermaßen dargestellt: Sin(90DEG).

△

5.1.9 Variablen

Ein Variablenname darf aus bis zu 16 Zeichen bestehen. Die folgenden Variablenwerte können eingesetzt werden. Sie unterscheiden sich anhand der Daten, die in ihnen abgelegt werden.

- Numerische Variablen, Zeichenkettenvariablen, Positionsvariablen, Gelenkvariablen, Ein- und Ausgabevariablen
- Numerische Variablen lassen sich weiterhin in ganze Zahlen (Integer), reelle Zahlen (Real) mit einfacher Genauigkeit und reelle Zahlen mit doppelter Genauigkeit unterteilen.

Variablen können nach ihrem Verwendungsbereich in lokale und globale Variablen eingeteilt werden.

- Lokale Variablen werden innerhalb eines Programms verwendet.
- Roboterstatusvariablen, externe Variablen und benutzerdefinierte externe Variablen sind globale Variablen und können auch programmübergreifend verwendet werden. (Benutzerdefinierte externe Variablen sind durch einen Unterstrich (_) als zweites Zeichen gekennzeichnet.)

Folgende Abbildung zeigt die verschiedenen Variablentypen:

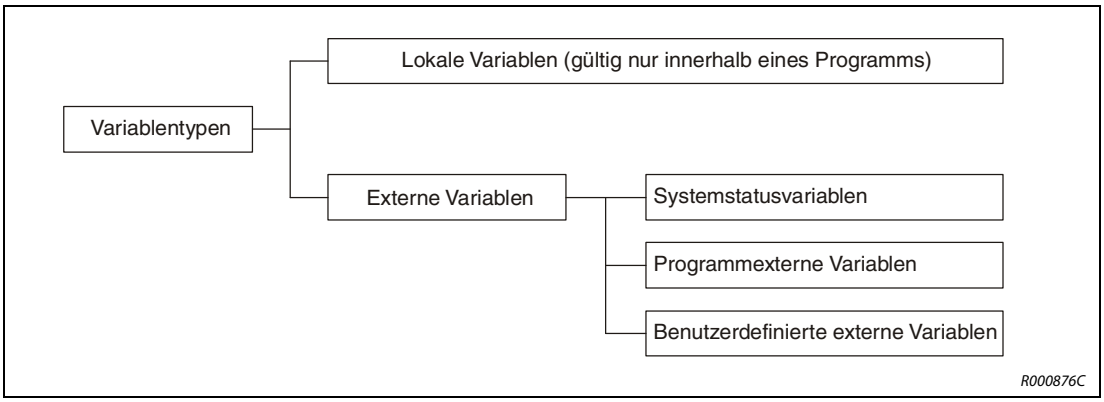


Abb. 5-8: *Variablentypen*

Folgende Abbildung zeigt die Syntax von Variablen:

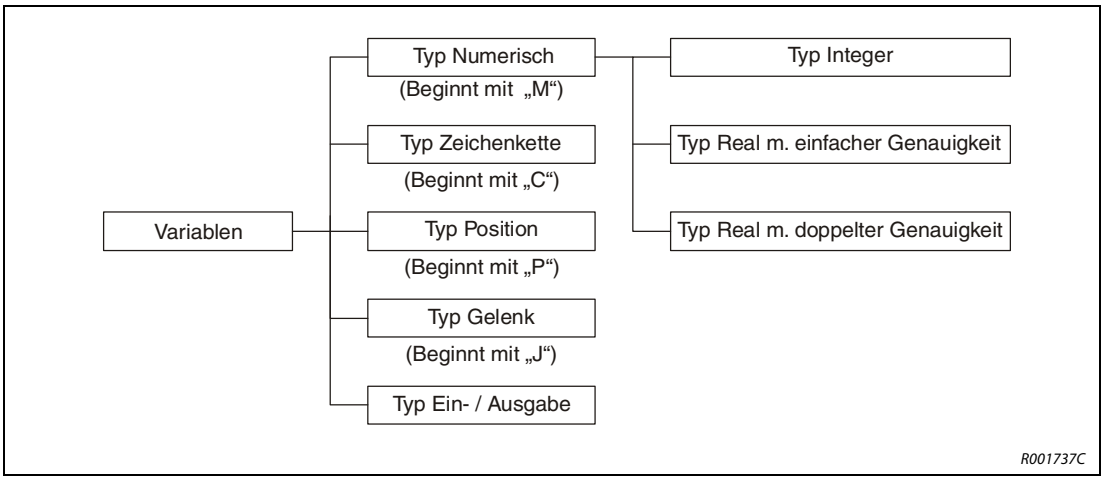


Abb. 5-9: *Variablen*

HINWEIS Variablen werden beim Generieren und beim Laden oder Zurücksetzen eines Programms nicht gelöscht.

Numerische Variablen

Variablen, deren Namen mit einem anderen Zeichen als mit „P“, „J“ oder „C“ beginnen, sind numerische Variablen. In MELFA-BASIC V wird eine numerische Variable in Anlehnung an den Anfangsbuchstaben des Wortes „Mathematik“ oft durch ein „M“ als erstes Zeichen im Variablennamen gekennzeichnet.

Beispiele▽

M1 = 100
M2! = −1.73E+10
M3# = 0.123
ABC = 1

△

Es besteht die Möglichkeit, den Typ einer numerischen Variablen durch ein Zeichen am Ende des Variablennamens zu kennzeichnen. Fehlt das Suffix in der Variablendeklaration, so wird die Variable als reelle Zahl mit einfacher Genauigkeit interpretiert.

Suffixe numerischer Variablen	Variablentyp
%	Integer
!	Real mit einfacher Genauigkeit
#	Real mit doppelter Genauigkeit

Tab. 5-2: *Typen von numerischen Variablen*

HINWEISE

Ein einmal registrierter Variablentyp kann nur vom Typ Integer in den Typ Real mit einfacher Genauigkeit umgewandelt werden. Es ist z.B. nicht möglich, eine Variable vom Typ Integer in den Typ Real mit doppelter Genauigkeit oder eine Variable vom Typ Real mit einfacher Genauigkeit in den Typ Real mit doppelter Genauigkeit umzuwandeln.

Es ist nicht möglich, zu einer bereits deklarierten Variablen ein Suffix hinzuzufügen. Das Suffix muss bei der Variablendeklaration während der Programmerstellung festgelegt werden.

Werden Zahlen mit einfacher Genauigkeit und Zahlen mit doppelter Genauigkeit gemeinsam verarbeitet oder ersetzt, tritt ein Fehler auf.

Typ	Bereich
Integer	−32 768 bis 32 767
Real mit einfacher Genauigkeit	−3,40282347E+38 bis 3,40282347E+38
Real mit doppelter Genauigkeit	−1,7976931348623157E+308 bis 1,7976931348623157E+308

Tab. 5-3: *Wertebereich*

Zeichenkettenvariablen

Der Name einer Zeichenkettenvariablen sollte mit „C“ beginnen und mit „$“ enden:

Beispiele▽

```
C1$ = "ABC"
CS$ = C1$
Def Char MOJI
MOJI = "MOJIMOJI"
```

△

Positionsvariablen

Der Name einer Positionsvariablen sollte mit „P“ beginnen. Es können jedoch auch andere Buchstaben verwendet werden. Die Deklaration einer Positionsvariablen erfolgt über die Anweisung Def Pos. Es ist möglich, auf einzelne Komponenten einer Positionsvariablen zuzugreifen. In diesem Fall muss nach dem Variablennamen ein Punkt „.“ gesetzt und der Name der Komponente (z.B. „X“) angegeben werden:

P1.X, P1.Y, P1.Z, P1.A, P1.B, P1.C, P1.L1, P1.L2

Die Einheit der Winkelkomponenten (z. B. P1.A) ist RAD. Verwenden Sie zur Umwandlung der Winkelkomponente in Grad die Funktion DEG.

Beispiele▽

```
P1 = PORG
Dim P3(10)
M1 = P1.X          (Einheit: mm)
M2 = Deg(P1.A)     (Einheit: Grad)
Def Pos L10
Mov L10
```

△

Gelenkvariablen

Der Name einer Gelenkvariablen sollte mit „J“ beginnen. Es können jedoch auch andere Buchstaben verwendet werden. Die Deklaration einer Gelenkvariablen erfolgt über die Anweisung Def Jnt. Es ist möglich, auf einzelne Komponenten einer Gelenkvariablen zuzugreifen. In diesem Fall muss nach dem Variablennamen ein Punkt „.“ gesetzt und der Name der Komponente (z.B. „J1“) angegeben werden:

JDATA.J1, JDATA.J2, JDATA.J3, JDATA.J4, JDATA.J5, JDATA.J6, JDATA.J7, JDATA.J8

Beispiele▽

```
JSTART = (0,0,90,0,90,0,0,0)
JDATA = JSTART
Dim J3(10)
M1 = J1.J1         (Einheit: Radiant)
M2 = Deg(J1.J2)    (Einheit: Grad)
Def Jnt K10
Mov K10
```

△

E/A-Variablen

Folgende Ein- und Ausgangsvariablen können verwendet werden. Sie sind als Roboterstatusvariablen vordefiniert.

E/A-Variable	Beschreibung
M_In	Für Lese-Zugriff auf Eingangssignalbits
M_Inb	Für Lese-Zugriff auf Eingangssignalbytes (8-Bit-Signal)
M_Inw	Für Lese-Zugriff auf Eingangssignalworte (16-Bit-Signal)
M_Out	Für Schreib-/Lese-Zugriff auf Ausgangssignalbits
M_OutB	Für Schreib-/Lese-Zugriff auf Ausgangssignalbytes (8-Bit-Signal)
M_OutW	Für Schreib-/Lese-Zugriff auf Ausgangssignalworte (16-Bit-Signal)
M_DIn	Für Lese-Zugriff auf Eingangsregister bei CC-Link-Verbindung (nicht bei der CRQ-Serie)
M_DOut	Für Schreib-/Lese-Zugriff auf Ausgangsregister bei CC-Link-Verbindung (nicht bei der CRQ-Serie)

Tab. 5-4: *E/A-Variablen*

Detaillierte Hinweise zu den E/A-Variablen finden Sie im Abschn. 7.2.28 „M_In/M_Inb/M_In8/M_Inw/M_In16“, im Abschn. 7.2.36 „M_Out/M_Outb/M_Out8/M_Outw/M_Out16“ und im Abschn. 7.2.22 „M_DIn/M_DOut“.

Feldvariablen

Numerische Variablen, Zeichenkettenvariablen, Positionsvariablen und Gelenkvariablen können in Feldvariablen verwendet werden. Die Festlegung der Feldvariablenelemente erfolgt über die Indizierung der jeweiligen Variablen. Deklarieren Sie die Feldvariablen, bevor Sie sie verwenden (siehe auch Abschn. 6.3.28 „Dim-Befehl“). Eine Feldvariable darf aus maximal 3 Dimensionen bestehen.

Beispiele▽

```
Dim M1(10)          Typ Real mit einfacher Genauigkeit
Dim M2%(10)         Typ Integer
Dim M3!(10)         Typ Real mit einfacher Genauigkeit
Dim M4#(10)         Typ Real mit doppelter Genauigkeit
Dim P1(20)
Dim J1(5)
Dim ABC (10,10,10)
```

△

Die Indizierung einer Variablen beginnt mit 1. Nur bei den speziellen Ein-/Ausgangssignalvariablen (M_In, M_Out etc.) beginnt die Indizierung bei 0. Stellen Sie vor der Deklaration von Feldvariablen sicher, dass genügend freier Speicherplatz vorhanden ist.

5.1.10 Externe Variablen

Externe Variablen sind durch einen Unterstrich (_) an der zweiten Stellen des Bezeichners (Variablenname) gekennzeichnet. Benutzerdefinierte externe Variablen müssen im Benutzerbasisprogramm registriert sein. Die Werte können programmübergreifend verwendet werden. Externe Variablen ermöglichen somit den Austausch von Daten zwischen Programmen.

Man unterscheidet vier Typen von externen Variablen:

- numerische Variablen
- Positionsvariablen
- Gelenkvariablen
- Zeichenkettenvariablen

Folgende externe Variablen stehen zur Verfügung:

Externe Variablen	Beschreibung	Beispiel
Programmexterne Variablen	Typ externer Variablen	P_01, M_01, P_100(1) usw.
Benutzerdefinierte externe Variablen	Der Name kann vom Benutzer frei definiert werden. Die Variablendeklaration erfolgt über Def Pos, Def Jnt, Def Char oder Def Inte/Float/Double im Benutzerbasisprogramm	P_GENTEN, M_MACHI
Roboterstatusvariablen	Roboterstatusvariablen sind Systemvariablen mit festgelegten Funktionen.	M_In, M_Out, P_Curr, M_PI usw.

Tab. 5-5: *Externe Variablen*

Programmexterne Variablen

In folgender Tabelle sind die verfügbaren programmexternen Variablen aufgeführt. Die Variablennamen sind festgelegt, die Anwendung kann vom Anwender definiert werden.

Datentyp	Variablenname	Anzahl
Position	P_00 bis P_19 P_20 bis P_39 ①	20 20
Positions-Feldvariable (Anzahl der Elemente: 10)	P_100( ) bis P_104( ) P_105( ) bis P_109( ) ①	5 5
Gelenk	J_00 bis J_19 J_20 bis J_39 ①	20 20
Gelenk-Feldvariable (Anzahl der Elemente: 10)	J_100( ) bis J_104( ) J_105( ) bis J_109( ) ①	5 5
Numerisch	M_00 bis M_19 M_20 bis M_39 ①	20 20
Numerische Feldvariable (Anzahl der Elemente: 10)	M_100( ) bis M_104( ) M_105( ) bis M_109( ) ①	5 5
Zeichenketten	C_00 bis C_19 C_20 bis C_39 ①	20 20
Zeichenketten-Feldvariablen (Anzahl der Elemente: 10)	C_100( ) bis C_104( ) C_105( ) bis C_109( ) ①	5 5

Tab. 5-6: *Programmexterne Variablen*

① Zur Nutzung des erweiterten Bereichs ist Parameter PRGGBL von „0 (Standard/Grundeinstellung)“ auf „1 (erweiterter Bereich)“ zu ändern und die Spannungsversorgung aus- und wieder einzuschalten. Wurde für eine programmexterne und eine benutzerdefinierte externe Variable zweimal derselbe Name vergeben, erfolgt beim Einschalten der Spannungsversorgung eine Fehlermeldung und eine Erweiterung des Bereichs ist nicht möglich. Ändern Sie in diesem Fall den Namen der benutzerdefinierten Variablen.

Benutzerdefinierte externe Variablen

Reicht die oben aufgelistete Anzahl an programmexternen Variablen nicht aus oder sollen spezielle Namen vergeben werden, können Sie im Benutzerbasisprogramm externe Variablen definieren.

Bevor Sie benutzerdefinierte externe Variablen verwenden können, müssen Sie:

- Ein Benutzerbasisprogramm erstellen
- Das Programm im Parameter „PRGUSR" registrieren und die Versorgungsspannung aus- und einschalten.
- Ein Programm zur Verwendung der benutzerdefinierten externen Variablen erstellen.

Eine über den Befehl Def deklarierte Variable, mit einem Unterstrich (_) an der zweiten Stelle im Variablennamen, wird als externe Variable verarbeitet. Dazu muss die Variable im Basisprogramm deklariert sein. Das Benutzerbasisprogramm muss nicht ausgeführt werden. Es reicht aus, nur die Zeilen mit den Variablendeklarationen zu erstellen.

Sollen im Benutzerbasisprogramm Feldvariablen erstellt und als externe Variablen eingesetzt werden, so ist eine zweite Deklaration über die Dim-Anweisung in dem Programm erforderlich, in dem sie verwendet werden. Lokale Variablen (Variablen, die nur innerhalb eines Programms verwendet werden) erfordern keine erneute Deklaration.

Folgendes Beispiel zeigt die Verwendung benutzerdefinierter externer Variablen.

Beispiel▽

Hauptprogramm (Programm 1)

```
1  Dim P_200(10)                'Zweite Deklaration der externen Variablen
2  Dim M_200(10)                'Zweite Deklaration der externen Variablen
3  Mov P_100(1)
4  If M_200(1) = 1 Then Hlt
5  M1 = 1                       'Lokale Variable
```

Benutzerbasisprogramm (Programm UBP)

```
1  Def Pos P_900, P_901, P_903  'Zweite Deklaration der externen Variablen
2  Dim P_200(10)                'Die Variable muss in dem Programm, in dem sie
                                 verwendet wird, noch einmal deklariert werden.
3  Def Inte M_100
4  Dim M_200(10)                'Die Variable muss in dem Programm, in dem sie
                                 verwendet wird, noch einmal deklariert werden.
```

Der Parameter „PRGUSR" muss auf „UBP" gesetzt werden.

△

Erstellung eines Benutzerbasisprogramms

Bei Verwendung von benutzerdefinierten Variablen wird ein Benutzerbasisprogramm benötigt. Dieses Programm wird nicht ausgeführt. Das Benutzerbasisprogramm enthält die Deklaration der benutzerdefinierten Variablen und wird über den Parameter PRGUSR definiert.

Nach Einstellung des Parameters muss die Versorgungsspannung aus- und wieder eingeschaltet werden.

Die Registrierung des Benutzerbasisprogramms erfolgt über die Software RT ToolBox2. Bei Verwendung der Software müssen lediglich die Anweisungen und anschließend die Positionsdaten festgelegt werden. Ein Benutzerbasisprogramm kann über die Teaching Box oder einen Personalcomputer erstellt werden. Gehen Sie bei Verwendung der Software RT ToolBox2 wie folgt vor:

① Speichern Sie ein als Benutzerbasisprogramm erstelltes Programm auf Ihren Personalcomputer.

② Starten Sie den Programm-Manager über den Programm-Editor.

③ Legen Sie im Programm-Manager das in ① erstellte Programm als Quelle und den Roboter als Ziel fest. Führen Sie einen Kopiervorgang aus. Achten Sie darauf, dass das Kontrollkästchen „Position Variables“ deaktiviert und das Kontrollkästchen „Instructions“ aktiviert ist.

④ Wiederholen Sie die unter ③ aufgeführten Anweisungen nach Abschluss des Kopiervorgangs. Führen Sie den Kopiervorgang jedoch mit aktiviertem Kontrollkästchen „Position Variables“ und deaktiviertem Kontrollkästchen „Instructions“ aus.

⑤ Löschen Sie zuerst das alte Benutzerbasisprogramm und speichern Sie das neue Programm mit Hilfe des Programm-Managers.

Roboterstatusvariablen

Roboterstatusvariablen werden verwendet, um einen schnellen Zugriff auf den Roboterzustand zu ermöglichen oder um ihn zu ändern. Die Variablennamen und die Funktion der Roboterstatusvariablen sind vordefiniert. Eine detaillierte Funktionsbeschreibung der Variablen finden Sie in Kap. 7.

Nr.	Variablen-name	Feldelement ①	Inhalt	Zugriff	Datentyp, Einheit	Seite
1	P_Curr	Mechanismus-nummer (1–3)	Augenblicksposition (XYZ)	Lesen	Position	7-66
2	J_Curr	Mechanismus-nummer (1–3)	Augenblicksposition (Gelenk)	Lesen	Gelenk	7-9
3	J_ECurr	Mechanismus-nummer (1–3)	Aktuelle Encoderposition	Lesen	Gelenk	7-9
4	J_Fbc	Mechanismus-nummer (1–3)	Gelenkposition abgeleitet aus der Servorückmeldung	Lesen	Gelenk	7-10
5	J_AmpFbc	Mechanismus-nummer (1–3)	Aktueller Wert der Servorückmeldung	Lesen	Gelenk	7-10
6	P_Fbc	Mechanismus-nummer (1–3)	XYZ-Position abgeleitet aus der Servorückmeldung	Lesen	Position	7-67
7	M_Fbd	Mechanismus-nummer (1–3)	Differenz zwischen der durch den Befehlswert vorgegebenen Sollpostion und der durch die Encoderimpulse gemeldeten Istposition	Lesen	Position	7-24
8	M_CmpDst	Mechanismus-nummer (1–3)	Abweichung zwischen Befehlswert und aktueller Position	Lesen	Real mit einfacher Genauigkeit, mm	7-16
9	M_CmpLmt	Mechanismus-nummer (1–3)	Zurücksetzen des Fehlerstatus durch Aufruf eines Interruptprozesses, wenn nach Aktivierung der Achsenweichheit ein Grenzwert überschritten wird	Lesen	Integer	7-17
10	P_Tool	Mechanismus-nummer (1–3)	Zuletzt festgelegte Werkzeugkonvertierungsdaten	Lesen	Position	7-69
11	P_Base	Mechanismus-nummer (1–3)	Zuletzt festgelegte Basiskonvertierungsdaten	Lesen	Position	7-63
12	P_NTool	Mechanismus-nummer (1–3)	Standardwert der Werkzeugkonvertierungsdaten	Lesen	Position	7-69
13	P_NBase	Mechanismus-nummer (1–3)	Standardwert der Basiskonvertierungsdaten	Lesen	Position	7-63
14	M_Tool	Mechanismus-nummer (1–3)	Zuletzt festgelegte Werkzeugnummer	Lesen	Integer	7-54
15	J_ColMxl	Mechanismus-nummer (1–3)	Maximale Drehmomentabweichung bei aktivierter Kollisionsüberwachung	Lesen	Gelenk	7-7
16	M_ColSts	Mechanismus-nummer (1–3)	Status der Kollisionsüberwachung	Lesen	Integer	7-18
17	P_ColDir	Mechanismus-nummer (1–3)	Bewegungsrichtung vor einem Zusammenstoß	Lesen	Position	7-64
18	M_OpOvrd	—	Geschwindigkeitsübersteuerung über das Bedienfeld des Steuergerätes (0–100 %)	Lesen	Integer, %	7-30
19	M_Ovrd	Programmplatz-nummer (1–32)	Zuletzt festgelegter Übersteuerungswert/gültig für das gesamte Programm (0–100 %)	Lesen	Integer, %	7-30

Tab. 5-7: *Roboterstatusvariablen (1)*

Nr.	Variablen-name	Feldelement ①	Inhalt	Zugriff	Datentyp, Einheit	Seite
20	M_JOvrd	Programmplatz-nummer (1–32)	Zuletzt festgelegter Übersteue-rungswert/gültig nur bei Gelenk-Interpolation (0–100 %)	Lesen	Integer, %	7-30
21	M_NOvrd	Programmplatz-nummer (1–32)	Systemstandardwert (M_Ovrd Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-30
22	M_NJovrd	Programmplatz-nummer (1–32)	Systemstandardwert (M_JOvrd Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-30
23	M_WupoV	Mechanismus-nummer (1–3)	Übersteuerung im Warmlaufbe-trieb (50–100 %)	Lesen	Real mit einfacher Genauigkeit, %	7-60
24	M_Wuprt	Mechanismus-nummer (1–3)	Restzeit einer Achse im Warmlauf-betrieb (s)	Lesen	Real mit einfacher Genauigkeit, s	7-61
25	M_Wupst	Mechanismus-nummer (1–3)	Zeit bis zur Wiederholung des Warmlaufbetriebs (s)	Lesen	Real mit einfacher Genauigkeit, s	7-62
26	M_Ratio	Programmplatz-nummer (1–32)	Position bezogen auf die Zielposi-tion	Lesen	Integer, %	7-45
27	M_RDst	Programmplatz-nummer (1–32)	Restverfahrweg zur Zielposition der Komponenten X, Y und Z	Lesen	Real mit einfacher Genauigkeit, mm	7-46
28	M_Spd	Programmplatz-nummer (1–32)	Zuletzt festgelegter Geschwindig-keitswert (gültig für Linear und Kreis-Interpolation)	Lesen	Real mit einfacher Genauigkeit, mm/s	7-51
29	M_NSpd	Programmplatz-nummer (1–32)	Systemstandardwert (M_Spd Standardwert)	Lesen	Real mit einfacher Genauigkeit, mm/s	7-51
30	M_RSpd	Programmplatz-nummer (1–32)	Aktuelle Geschwindigkeit (gültig für Linear- und Kreis-Inter-polation)	Lesen	Real mit einfacher Genauigkeit, mm/s	7-51
31	M_Acl	Programmplatz-nummer (1–32)	Zuletzt festgelegte Beschleuni-gungszeit	Lesen	Real mit einfacher Genauigkeit, %	7-12
32	M_DAcl	Programmplatz-nummer (1–32)	Zuletzt festgelegte Abbremszeit	Lesen	Real mit einfacher Genauigkeit, %	7-12
33	M_NAcl	Programmplatz-nummer (1–32)	Systemstandardwert (M_Acl Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-12
34	M_NDAcl	Programmplatz-nummer (1–32)	Systemstandardwert (M_DAcl Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-12
35	M_AclSts	Programmplatz-nummer (1–32)	Aktueller Status der Beschleuni-gung-/Abbremsung 0 = gestoppt, 1 = beschleunigt, 2 = konstante Geschwindigkeit, 3 = bremst	Lesen	Integer	7-12
36	M_SetAdl	Achsennummer (1–8)	Verhältnis der Beschleunigungs/ Abbremszeiten	Lesen/ schreiben	Real mit einfacher Genauigkeit, %	7-48
37	M_LdFact	Achsennummer (1–8)	Lastverhältnis der einzelnen Ser-vomotorachsen	Lesen	Real mit einfacher Genauigkeit, %	7-31
38	M_Run	Programmplatz-nummer (1–32)	Programmstatus 1 = Betrieb, 0 = kein Betrieb	Lesen	Integer	7-47

Tab. 5-7: *Roboterstatusvariablen (2)*

Nr.	Variablen-name	Feldelement ①	Inhalt	Zugriff	Datentyp, Einheit	Seite
39	M_Wai	Programmplatz-nummer (1–32)	Wartestatus 1 = Pause, 0 = keine Pause	Lesen	Integer	7-59
40	M_Psa	Programmplatz-nummer (1–32)	Status der Programm-wählbarkeit 1 = Auswahl freigegeben 0 = Auswahl gesperrt (Pause)	Lesen	Integer	7-44
41	M_Cys	Programmplatz-nummer (1–32)	Zyklusbetrieb aktiv 1 = Zyklusbetrieb 0 = kein Zyklusbetrieb	Lesen	Integer	7-19
42	M_CStp	—	Zyklusstopp 1 = Betrieb, 0 = kein Betrieb	Lesen	Integer	7-19
43	C_Prg	Programmplatz-nummer (1–32)	Programmname für Ausführung	Lesen	Zeichenkette	7-5
44	M_Line	Programmplatz-nummer (1–32)	Aktuell ausgeführte Zeilennummer	Lesen	Integer	7-33
45	M_SkipCq	Programmplatz-nummer (1–32)	Wird nach Ausführung eines SKIP-Befehls auf „1“, sonst auf „0“ gesetzt	Lesen	Integer	7-50
46	M_BrkCq	—	Ergebnis der Break-Befehlsaus-führung (1 = Bremsen aktiviert, 0 = Bremsen gelöst)	Lesen	Integer	7-14
47	M_Err	—	Fehlermeldung 1 = Fehler, 0 = kein Fehler	Lesen	Integer	7-21
48	M_ErrLvl	—	Schwelle für Lesefehler: ● SW-Version R1c oder früher (SQ-Serie)/S1c oder früher (SD-Serie) kein Fehler/Warnung/niedrig/hoch = 0/1/2/3 ● SW-Version R1d oder später (SQ-Serie)/S1d oder später (SD-Serie) kein Fehler/Warnung/niedrig/hoch/Warnung1/niedrig1/hoch1 = 0/1/2/3/4/5/6	Lesen	Integer	7-21
49	M_Errno	—	Fehlernummer lesen	Lesen	Integer	7-21
50	M_Svo	Mechanismus-nummer (1–3)	Servospannung EIN 1 = Servo EIN, 0 = Servo AUS	Lesen	Integer	7-52
51	M_Uar	Mechanismus-nummer (1–3)	Bitdaten 1 = innerhalb des benutzer-definierten Bereiches, 0 = außerhalb des benutzer-definierten Bereiches; Bit 0: Bereich 1 ... Bit 7: Bereich 8	Lesen	Integer	7-56
52	M_Uar32	Mechanismus-nummer (1–3)	Bitdaten 1 = innerhalb des benutzer-definierten Bereiches, 0 = außerhalb des benutzer-definierten Bereiches; Bit 0: Bereich 1 ... Bit 31: Bereich 32	Lesen	Integer	7-57
53	M_In	Eingangs-nummer (0–32767)	Verwenden Sie die Variable zur Eingabe externer Bitsignale Allgemeine Bit-Schnittstelle: Bit-Eingang: 0 = AUS, 1 = EIN Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26
54	M_INB/ M_In8	Eingangs-nummer (0–32767)	Verwenden Sie die Variable zur Eingabe externer Bytesignale (8 Bit) Allgemeine Bit-Schnittstelle: Byte-Eingang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26

Tab. 5-7: *Roboterstatusvariablen (3)*

Nr.	Variablen-name	Feldelement ①	Inhalt	Zugriff	Datentyp, Einheit	Seite
55	M_INW/ M_In16	Eingangs-nummer (0–32767)	Verwenden Sie die Variable zur Eingabe externer Bytesignale (16 Bit) Allgemeine Bit-Schnittstelle: Wort-Eingang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26
56	M_In32	Eingangs-nummer (0–32767)	Verwenden Sie die Variable zur numerischen Eingabe externer Bytesignale (32 Bit) Allgemeine Bit-Schnittstelle: Doppelwort-Eingang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26
57	M_Out	Ausgangs-nummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bitsignale Allgemeine Bit-Schnittstelle: Bit-Ausgang: 0 = AUS, 1 = EIN Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ schreiben	Integer	7-37
58	M_OutB/ M_Out8	Ausgangs-nummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bytesignale (8 Bit) Allgemeine Bit-Schnittstelle: Byte-Ausgang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ schreiben	Integer	7-37
59	M_OutW/ M_Out16	Ausgangs-nummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bytesignale (16 Bit) Allgemeine Bit-Schnittstelle: Wort-Ausgang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ schreiben	Zeichenkette	7-37
60	M_Out32	Ausgangs-nummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bytesignale (32 Bit) Allgemeine Bit-Schnittstelle: Doppelwort-Ausgang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ schreiben	Integer	7-40
61	M_DIn	Eingangs-nummer (ab 6000)	Dezentrales CC-Link-Register: Eingangsregister (nicht bei der CRnQ-Serie)	Lesen	Integer	7-20
62	M_DOut	Ausgangs-nummer (ab 6000)	Dezentrales CC-Link-Register: Ausgangsregister (nicht bei der CRnQ-Serie)	Lesen/ schreiben	Integer	7-20
63	M_HndCq	Eingangs-nummer (1–8)	Eingang Handgreiferzustand	Lesen	Integer	7-25
64	P_Safe	Mechanismus-nummer (1–3)	Position des Rückzugspunktes	Lesen	Position	7-68
65	J_Origin	Mechanismus-nummer (1–3)	Gelenkkoordinaten des Referenz-punktes	Lesen	Gelenk	7-11
66	M_Open	Dateinummer (1–8)	Prüft, ob eine Datei oder Kommu-nikationsleitung geöffnet ist	Lesen	Integer	7-35
67	C_Mecha	Programmplatz-nummer (1–32)	Name des Mechanismus	Lesen	Zeichenkette	7-5
68	C_Maker	—	Herstellerinformation (max. 64 Zeichen)	Lesen	Zeichenkette	7-4
69	C_User	—	Inhalt des Parameters „USERMSG“ (max. 64 Zeichen)	Lesen	Zeichenkette	7-6
70	C_Date	—	Aktuelles Datum Jahr/Monat/Tag	Lesen	Zeichenkette	7-4
71	C_Time	—	Aktuelle Zeit Stunde/Minute/Sekunde	Lesen	Zeichenkette	7-6

Tab. 5-7: *Roboterstatusvariablen (4)*

Nr.	Variablen-name	Feldelement ①	Inhalt	Zugriff	Datentyp, Einheit	Seite
72	M_BTime	—	Restzeit der Batterie	Lesen	Integer, Stunden	7-15
73	M_Timer	Timernummer (1–8)	Zeitdauer ab Bezugszeit	Lesen/ schreiben	Real mit einfacher Genauigkeit, ms	7-53
74	P_Zero	—	Variable, deren Komponenten (X, Y, Z, A, B, C, FL1, FL2) alle auf „0“ gesetzt sind.	Lesen	Position	7-72
75	M_PI	—	Kreiszahl (3.1415...)	Lesen	Real mit doppelter Genauigkeit	7-44
76	M_Exp	—	Basis des natürlichen Logarithmus (2.71828...)	Lesen	Real mit doppelter Genauigkeit	7-23
77	M_G	—	Erdbeschleunigung (9.80665)	Lesen	Real mit doppelter Genauigkeit	7-25
78	M_On	—	Eine „1“ wird gesetzt.	Lesen	Integer	7-34
79	M_Off	—	Eine „0“ wird gesetzt.	Lesen	Integer	7-34
80	M_Mode	—	Betriebsmoduseinstellung des Drehschalters am Steuergerät: MANUAL = 1, AUTOMATIC = 2	Lesen	Integer	7-34

Tab. 5-7: *Roboterstatusvariablen (5)*

① Mechanismusnummer: Festlegung der Mechanismusnummer (1–3) im Multitasking-Betrieb
Programmplatznummer: Festlegung des Programmplatzes (1–32) im Multitasking-Betrieb
Eingangsnummer: Bit-Nummer des Eingangssignals (0–32767)
Ausgangsnummer: Bit-Nummer des Ausgangssignals (0–32767)

Logische Werte

Logische Werte geben die Resultate von Vergleichsoperationen oder Ein- und Ausgangszuständen wieder. Ein Ergebnis ungleich 0 entspricht dem Wert „wahr“ und ein Ergebnis gleich 0 entspricht dem Wert „unwahr“. Ergebnisse werden im Integer-Format angezeigt. Bei Substitutionen wird für den Wert „wahr“ eine „1“ gesetzt. Folgende Tabelle zeigt die logischen Werte und deren Bedeutung:

Durch eine „1“ dargestellte Zustände	Durch eine „0“ dargestellte Zustände
Ergebnis einer Vergleichsoperation (falls wahr)	Ergebnis einer Vergleichsoperation (falls unwahr)
Ergebnis einer logischen Operation (falls wahr)	Ergebnis einer logischen Operation (falls unwahr)
Schalter EIN	Schalter AUS
Ein-/Ausgangsignal EIN	Ein-/Ausgangsignal AUS
Handgreifer offen (Stromfluss durch die Hand)	Handgreifer geschlossen (kein Stromfluss durch die Hand)
Einstellungen für Freigabe oder Gültigkeit (wie zum Beispiel bei Interrupts)	Einstellungen für Sperren oder Ungültigkeit (wie zum Beispiel bei Interrupts)

Tab. 5-8: *Logische Werte und deren Bedeutung*

5.1.11 Funktionen

Mit dem Argument einer Funktion wird eine durch die Funktion festgelegte Rechenoperation durchgeführt. Das Ergebnis kann ein numerischer Typ oder eine Zeichenkette sein.

Benutzerdefinierte Funktionen

Benutzerdefinierte Funktionen werden mit dem Befehl Def FN erstellt.

Beispiel▽

```
Def FNMADD(MA, MB) = MA + MB
```

△

Funktionen beginnen mit den Zeichen „FN". Das dritte Zeichen dient zur Beschreibung des Datentyps (Zeichenkette: C, Numerischer Wert: M, Position: P, Gelenk: J). Es können bis zu 8 Zeichen verwendet werden.

Fest definierte Funktionen

Folgende Tabelle zeigt die fest definierten Funktionen:

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Numerische Funktionen	Abs (<Numerischer Ausdruck>)	Bildet den Betrag	8-2	Numerischer Wert
	Cint (<Numerischer Ausdruck>)	Rundet den dezimalen Wert zu einer Integer-Zahl	8-9	
	Deg (<Numerischer Ausdruck: radian>)	Wandelt die Einheit des Winkels von Radiant (rad) in Grad (deg) um	8-13	
	Exp (<Numerischer Ausdruck>)	Berechnet den Wert der Exponentialfunktion	8-14	
	Fix (<Numerischer Ausdruck>)	Erzeugt einen Integer-Anteil	8-15	
	Int (<Numerischer Ausdruck>)	Erzeugt die größtmögliche Integer-Zahl, die kleiner als der Wert des numerischen Ausdrucks ist	8-18	
	Len (<Ausdruck für eine Zeichenkette>)	Berechnet die Länge der Zeichenkette	8-20	
	Ln (<Numerischer Ausdruck>)	Berechnet den natürlichen Logarithmus	8-21	
	Log (<Numerischer Ausdruck>)	Berechnet den dekadischen Logarithmus	8-21	
	Max (<Numerischer Ausdruck>...)	Berechnet den Maximalwert der numerischen Ausdrücke	8-22	
	Min (<Numerischer Ausdruck>...)	Berechnet den Minimalwert der numerischen Ausdrücke	8-24	
	Rad (<Numerischer Ausdruck: deg>)	Wandelt die Einheit des Winkels von Grad (deg) in Radiant (rad) um	8-30	
	Sgn (<Numerischer Ausdruck>)	Prüft das Vorzeichen des numerischen Ausdrucks	8-40	
	Sqr (<Numerischer Ausdruck>)	Berechnet die Quadratwurzel	8-41	
	Strpos (<Ausdruck für eine Zeichenkette>, <Ausdruck für eine Zeichenkette>)	Gibt die Position der zweiten Zeichenkette innerhalb der ersten Zeichenkette an	8-41	
	Rnd (<Numerischer Ausdruck>)	Ermittelt eine Zufallszahl	8-33	
Numerische Funktionen	Asc (<Typ Zeichenkette>)	Erzeugt den ASCII-Code für das erste Zeichen in der Zeichenkette	8-4	Numerischer Wert
	Cvi (<Typ Zeichenkette>)	Wandelt eine 2-Byte-Zeichenkette in einen Integer-Wert um	8-11	
	Cvs (<Typ Zeichenkette>)	Wandelt eine 4-Byte-Zeichenkette in einen Real-Wert mit einfacher Genauigkeit um	8-12	
	Cvd (<Typ Zeichenkette>)	Wandelt eine 8-Byte-Zeichenkette in einen Real-Wert mit doppelter Genauigkeit um	8-12	
	Val (<Typ Zeichenkette>)	Wandelt eine Zeichenkette in einen numerischen Wert um	8-43	

Tab. 5-9: *Fest definierte Funktionen (1)*

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Trigonometrische Funktionen	Atn (<Numerischer Ausdruck>)	Berechnet den Arcus Tangens (Einheit: rad) Definitionsbereich: numerischer Wert, $-\pi/2$ bis $+\pi/2$	8-5	Numerischer Wert
	Atn2 (<Numerischer Ausdruck>, <Numerischer Ausdruck>)	Berechnet den Arcus Tangens (Einheit: rad) (Θ = ATN2(Δy, Δx)) Definitionsbereich: numerischer Wert von Δy und Δx ungleich 0, $-\pi$ bis $+\pi$	8-5	
	Cos (<Numerischer Ausdruck>)	Berechnet den Kosinus (Einheit: rad) Definitionsbereich: numerischer Wert, -1 bis $+1$	8-11	
	Sin (<Numerischer Ausdruck>)	Berechnet den Sinus (Einheit: rad) Definitionsbereich: numerischer Wert, -1 bis $+1$	8-40	
	Tan (<Numerischer Ausdruck>)	Berechnet den Tangens (Einheit: rad) Definitionsbereich: numerischer Wertebereich	8-42	
Zeichenkettenfunktionen	Bin$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine binäre Zeichenkette um	8-6	Zeichenkette
	Chr$ (<Numerischer Ausdruck>)	Erzeugt ein Zeichen, das dem Wert des numerischen Ausdrucks entspricht	8-9	
	Hex$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine hexadezimale Zeichenkette um	8-17	
	Left$ (<Zeichenkette>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette Die Länge der erzeugten Zeichenkette, beginnend mit dem linken Zeichen, ist im zweiten Argument festgelegt.	8-20	
	Mid$ <Zeichenkette>, <Numerischer Ausdruck>, <Numerischer Ausdruck>	Erzeugt einen Teil der Zeichenkette Die Länge der erzeugten Zeichenkette ist im dritten, die Position von links im zweiten Argument festgelegt.	8-23	
	Mirror$ (<Typ Zeichenkette>)	Spiegelung der binären Bits der Zeichenkette	8-24	
	Mki$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine 2-Byte-Zeichenkette um	8-25	
	Mks$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine 4-Byte-Zeichenkette um	8-26	
	Mkd$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine 8-Byte-Zeichenkette um	8-27	
	Right$ (<Zeichenkette>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette Die Länge der erzeugten Zeichenkette, beginnend mit dem rechten Zeichen, ist im zweiten Argument festgelegt.	8-34	
	Str$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine dezimale Zeichenkette um	8-42	
	CkSum <Zeichenkette>, <Numerischer Ausdruck>, <Numerischer Ausdruck>	Bildet die Prüfsumme der Zeichenkette Schreibt den Wert des niederwertigen Bytes aus der Summe der im zweiten und dritten Argument festgelegten Zeichenkette in die Zeichenkette des ersten Arguments	8-10	Numerischer Wert

Tab. 5-9: *Fest definierte Funktionen (2)*

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Positions-variablen	Dist (<Position>, <Position>)	Berechnet den Abstand zwischen zwei Punkten	8-14	Position
	Fram (<Position 1>, <Position 2>, <Position 3>)	Berechnet das Koordinatensystem über 3 Punkte Position 1 entspricht dem Flächenursprung, Position 2 dem Punkt in der Fläche der X- und Position 3 dem Punkt in der Fläche der Y-Achse. Der Flächenursprungspunkt und die Stellung sind durch die X-, Y- und Z-Koordinaten der 3 Positionen beschrieben und können über den Rücksetzwert (Position) zurückgesetzt werden. Die Ausführung erfolgt ohne Berücksichtigung der Mechanismusstruktur über 6 Achsen und 3 Dimensionen. Die Funktion kann nicht bei 5-achsigen Robotern verwendet werden, da die Orientierungsdaten A, B und C eine andere Bedeutung haben.	8-16	
	Rdfl1 (<Position>, <Numerischer Wert>)	Überträgt den Stellungsmerker der festgelegten Position als Zeichenkette Argument <numerischer Wert>: 0 = R/L, 1 = A/B, 2 = F/N	8-31	Zeichen-kette
	Setfl1 (<Position>, <Zeichen>)	Änderung des Stellungsmerkers der festgelegten Position Die zu ändernden Daten werden über Zeichen definiert (R/L/A/B/F/N).	8-35	
	Rdfl2 (<Position>, <Numerischer Wert>)	Überträgt die Multirotationsdaten der festgelegten Position als numerischen Wert (−2 bis 1) Das Argument <numerischer Ausdruck> überträgt die Achsennummer (1 bis 8).	8-32	Numerischer Wert
	Setfl2 (<Position>, <Numerischer Wert>, <Numerischer Wert>)	Änderung der Multirotationsdaten der festgelegten Position als numerischer Wert (−2 bis 1) Die linke Seite des Ausdrucks entspricht der Achsennummer, die geändert werden soll, die rechte Seite entspricht dem Wert.	8-37	
	Align (<Position>)	Setzt den Wert der Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand zur Stellung (A, B, C) der Position 1 Die Funktion kann nicht bei 5-achsigen Robotern verwendet werden, da die Orientierungsdaten A, B und C eine andere Bedeutung haben.	8-3	
	Inv (<Position>)	Invertiert die Positions-Matrix	8-18	Position
	PtoJ (<Position>)	Konvertiert die Positions- in Gelenkdaten	8-29	Gelenk
	JtoP (<Position>)	Konvertiert die Gelenk- in Positionsdaten	8-19	Position
	Zone (<Position 1>, <Position 2>, <Position 3>)	Prüft, ob die Position 1 innerhalb des durch die Positionen 2 und 3 definierten Quaders liegt (außerhalb = 0, innerhalb = 1) Für Komponenten, die nicht geprüft werden sollen oder fehlen, müssen die entsprechenden Positionskoordinaten auf folgende Werte gesetzt werden: Ist die Einheit Grad, muss Position 2 auf −360° und Position 3 auf 360° gesetzt werden. Ist die Einheit mm, muss Position 2 auf −10000 und Position 3 auf 10000 gesetzt werden.	8-44	Numerischer Wert

Tab. 5-9: *Fest definierte Funktionen (3)*

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Positions-variablen	Zone2 (<Position 1>, <Position 2>, <Position 3>, <Numerischer Wert 1>, <Numerischer Wert 2>, <Numerischer Wert 3>, <Position 4>)	Prüft, ob die Position 1 innerhalb des durch die Positionen 2 und 3 definierten Zylinders liegt (außerhalb = 0, innerhalb = 1) Es werden nur die Koordinaten X, Y und Z geprüft; die Orientierungsdaten A, B und C werden ignoriert.	8-46	Numerischer Wert
	PosCq (<Position>)	Prüft, ob die Position innerhalb des gültigen Bewegungsbereiches liegt	8-27	Numerischer Wert
	PosMid (<Position 1>, <Position 2>, <Numerischer Wert 1>, <Numerischer Wert 2>)	Berechnet die mittlere Position zwischen <Position 1> und <Position 2>	8-28	Position
	CalArc (<Position 1>, <Position 2>, <Position 3>, <Numerischer Wert 1>, <Numerischer Wert 2>, <Numerischer Wert 3>, <Position 4>)	Enthält die Daten des über die <Position 1>, <Position 2> und <Position 3> definierten Kreises	8-7	Numerischer Wert
	SetJnt (<J1-Achse>, <J2-Achse>, <J3-Achse>, <J4-Achse>, <J5-Achse>, <J6-Achse>, <J7-Achse>, <J8-Achse>)	Ändert die Werte einer Gelenkvariablen	8-38	Gelenk
	SetPos (<X-Achse>, <Y-Achse>, <Z-Achse>, <A-Achse>, <B-Achse>, <C-Achse>, <L1-Achse>, <L2-Achse>)	Ändert die Werte einer Positionsvariablen	8-39	Position

Tab. 5-9: *Fest definierte Funktionen (4)*

5.1.12 Operanden

Numerische Variablen müssen in MELFA-BASIC V nicht als Typ Integer oder Real deklariert werden. In Abhängigkeit von der ausgeführten Operation werden die Daten automatisch konvertiert. Dabei kann das Ergebnis in Abhängigkeit von der Reihenfolge der Datentypen unterschiedlich sein. Folgende Tabelle zeigt einige Beispiele:

Linkes Argument	Operation	Rechtes Argument	Ergebnis
15 (Typ Numerisch)	AND	256 (Typ Numerisch)	15 (Typ Numerisch)
P1 (Typ Position)	*	M1 (Typ Numerisch)	P2 (Typ Position)
M1 (Typ Numerisch)	*	P1 (Typ Position)	FEHLER

Tab. 5-10: *Operationsergebnisse in Abhängigkeit der Datenreihenfolge*

Konvertierung der Datentypen in Abhängigkeit der Operation

Folgende Tabelle zeigt die Konvertierung von Datentypen in Abhängigkeit von der ausgeführten Operation. Bei der Angabe von logischen Operationen ist die logische Negation ausgenommen.

Typ linkes Argument		Operation	Typ rechtes Argument				
			Zeichenkette	Numerischer Wert		Position	Gelenk
				Integer	Real		
Zeichenkette		Substitution	Zeichenkette	—	—	—	—
		Addition	Zeichenkette	—	—	—	—
		Vergleich	Integer	—	—	—	—
Numerischer Wert	Integer	Addition	—	Integer	Real	—	—
		Subtraktion	—	Integer	Real	—	—
		Multiplikation	—	Integer	Real	—	—
		Division	—	Integer	Real	—	—
		Integer-Division	—	Integer	Integer	—	—
		Modulo	—	Integer	Integer	—	—
		Exponential	—	Integer	Integer	—	—
		Substitution	—	Integer	Integer	—	—
		Vergleich	—	Integer	Integer	—	—
		Logisch	—	Integer	Integer	—	—
	Real	Addition	—	Real	Real	—	—
		Subtraktion	—	Real	Real	—	—
		Multiplikation	—	Real	Real	—	—
		Division	—	Real	Real	—	—
		Integer-Division	—	Integer	Integer	—	—
		Modulo	—	Integer	Integer	—	—
		Exponential	—	Integer	Real	—	—
		Substitution	—	Integer	Real	—	—
		Vergleich	—	Integer	Integer	—	—
		Logisch	—	Integer	Integer	—	—

Tab. 5-11: *Konvertierung der Datentypen (1)*

Typ linkes Argument	Operation	Typ rechtes Argument				
		Zeichenkette	Numerischer Wert		Position	Gelenk
			Integer	Real		
Position	Addition	—	—	—	Position	—
	Subtraktion	—	—	—	Position	—
	Multiplikation	—	Position	Position	Position	—
	Division	—	Position	Position	Position	—
	Integer-Division	—	—	—	—	—
	Modulo	—	—	—	—	—
	Exponential	—	—	—	—	—
	Substitution	—	—	—	Position	—
	Vergleich	—	—	—	—	—
	Logisch	—	—	—	—	—
Gelenk	Addition	—	—	—	—	Gelenk
	Subtraktion	—	—	—	—	Gelenk
	Multiplikation	—	Gelenk	Gelenk	—	—
	Division	—	Gelenk	Gelenk	—	Gelenk
	Integer-Division	—	—	—	—	—
	Modulo	—	—	—	—	—
	Exponential	—	—	—	—	—
	Substitution	—	—	—	—	Gelenk
	Vergleich	—	—	—	—	—
	Logisch	—	—	—	—	—
Nur linkes Argument	Vorzeichenumkehr	—	Integer	Integer	Position	Gelenk
	Negation NOT	—	Integer	Integer	—	—

Tab. 5-11: *Konvertierung der Datentypen (2)*

HINWEISE

Eine Division durch „0“ ist nicht möglich.

Bei der Ausführung exponentieller, modulo und logischer Operationen werden reelle Zahlen vor der Verarbeitung in ganzzahlige Werte umgewandelt und abgerundet.

5.1.13 Rangfolge von Operationen

Werden in einem Ausdruck mehrere Operationen ausgeführt, gilt die in folgender Tabelle dargestellte Rangfolge:

Operation (Operator)	Typ der Operation	Priorität
Operation in Klammern ( )	—	Hoch
Funktion	Funktion	
Exponent (^)	Operation mit numerischen Daten	
Operation mit einem Argument (+, –)	Operation mit numerischen Daten	
* /	Operation mit numerischen Daten	
\	Operation mit numerischen Daten	
MOD	Operation mit numerischen Daten	
+ –	Operation mit numerischen Daten	
<< >>	Logische Operation	
Vergleichsoperation (=, <>, ><, <, <=, =<, >, >=, =>)	Vergleichsoperation	
NOT	Logische Operation	
AND	Logische Operation	
OR	Logische Operation	
XOR	Logische Operation	Niedrig

Tab. 5-12: *Rangfolge von Operationen*

5.1.14 Programmebenen

Beim Entwurf eines Programms muss die Anzahl der Ebenen und die Struktur festgelegt werden. Werden die in folgender Tabelle aufgeführten Befehle verwendet, erweitert sich die Programmstruktur um eine Ebene. Für jeden Befehl gibt es eine maximale Anzahl der Ebenen. Wird diese Anzahl überschritten, erfolgt eine Fehlermeldung.

Anzahl der Ebenen	Verfügbare Befehle
16 Ebenen	Wiederholschleifen (For ~ Next, While ~ WEnd)
8 Ebenen	Funktionsaufruf (CallP)
800 Ebenen	Unterprogrammaufruf (GoSub) ①

Tab. 5-13: *Programmebenen*

① Durch die Verwendung der Anweisungen For–Next, While–WEnd und CallP wird die Programmstruktur flacher.

5.1.15 Reservierte Wörter

Reservierte Wörter haben im System eine bestimmte, festliegende Bedeutung. Sie dürfen zum Beispiel nicht als Programmname etc. vergeben werden. Zu den reservierten Wörtern zählen z.B. Anweisungen, Funktionen und Systemstatusvariablen.

5.2 Koordinatensysteme des Roboters

5.2.1 Beschreibung der Koordinatensysteme

Der Roboter verfügt über vier Koordinatensysteme.

Bezeichnung	Bedeutung
Weltkoordinatensystem	Bezogen auf den Aufstellort Im Normalfall bezieht sich die aktuelle Position auf dieses Koordinatensystem (siehe HINWEIS in der Abb. 5-10).
Basiskoordinatensystem	Bezogen auf die Standfläche des Roboterarms Der „Standardbasiskoordinaten-Parameter“ (MEXBS) gibt die Relation zum Weltkoordinatensystem an, die durch die Basis-Konvertierungsdaten oder den Base-Befehl vorgegeben wurde. In der Werkseinstellung sind die Basiskonvertierungsdaten auf „0“ gesetzt, so dass das Weltkoordinatensystem dem Basiskoordinatensystem entspricht.
Handflanschkoordinatensystem	Bezogen auf den Handflansch
Werkzeugkoordinatensystem	Bezogen auf die Werkzeugspitze Die Abstandswerte zum Werkzeugkoordinatensystem sind durch die Werkzeugdaten (z.B. vorgegeben durch Parameter MEXTL oder den Tool-Befehl) definiert.

Tab. 5-14: *Bedeutung der Koordinatensysteme*

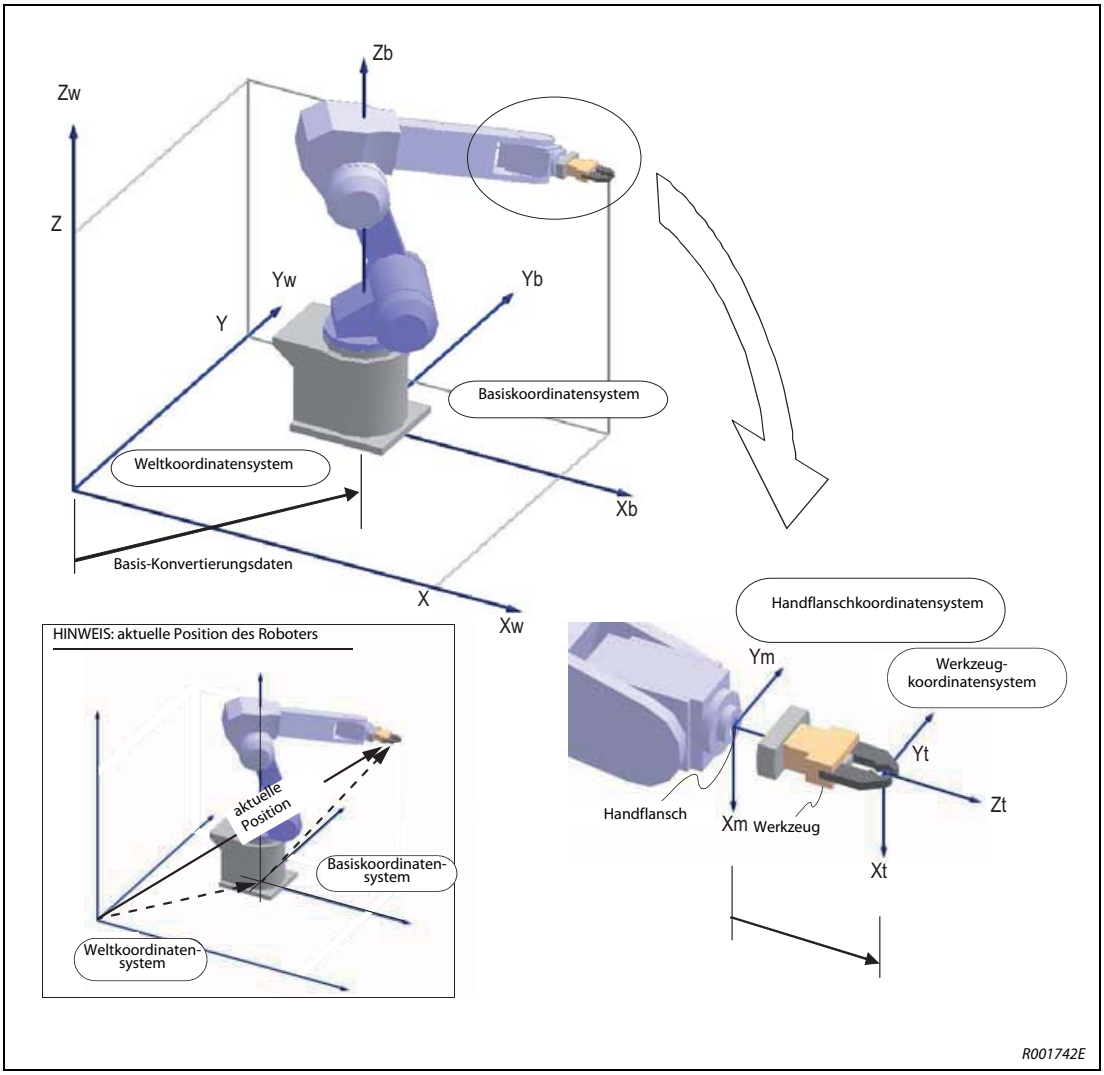


Abb. 5-10: *Koordinatensysteme des Roboters*

5.2.2 Basis-Konvertierung

Mit Hilfe der Basis-Konvertierung kann das Weltkoordinatensystem bewegt und in den Referenzpunkt des Werktisches oder Werkstückes gelegt werden. Die Darstellung der Roboterpositionen erfolgt dann in Bezug auf den Werktisch oder das Werkstück.
Die Funktion ist zum Beispiel dann sinnvoll, wenn der Roboter die gleichen Verfahrbewegungen an verschiedenen Stellen ausführen soll. Die einzelnen Positionen müssen dann nicht mehr für jede Stelle geteacht werden, sondern ergeben sich durch die Basis-Konvertierung.
Die Vorgabe der Daten für die Basis-Konvertierung erfolgt über den Parameter MEXBS oder den Base-Befehl.

Die Basis-Konvertierungsdaten beschreiben den Nullpunkt des Basiskoordinatensystems in Relation zu dem Nullpunkt des Weltkoordinatensystems, das durch die Konvertierung neu definiert wird. Möchten Sie daher die Daten unter Verwendung der aktuellen Position des Roboters festlegen (mit Hilfe der Fram-Funktion o.Ä.), führen Sie eine inverse Konvertierung der Positionsdaten durch (z.B. Base Inv(P1)).
Legen Sie die Parameter für das Werkstückkoordinatensystem WK1CRD bis WK3CRD über einen Basebefehl fest, müssen Sie die inverse Konvertierung nicht selbst durchführen, da diese intern ausgeführt wird.

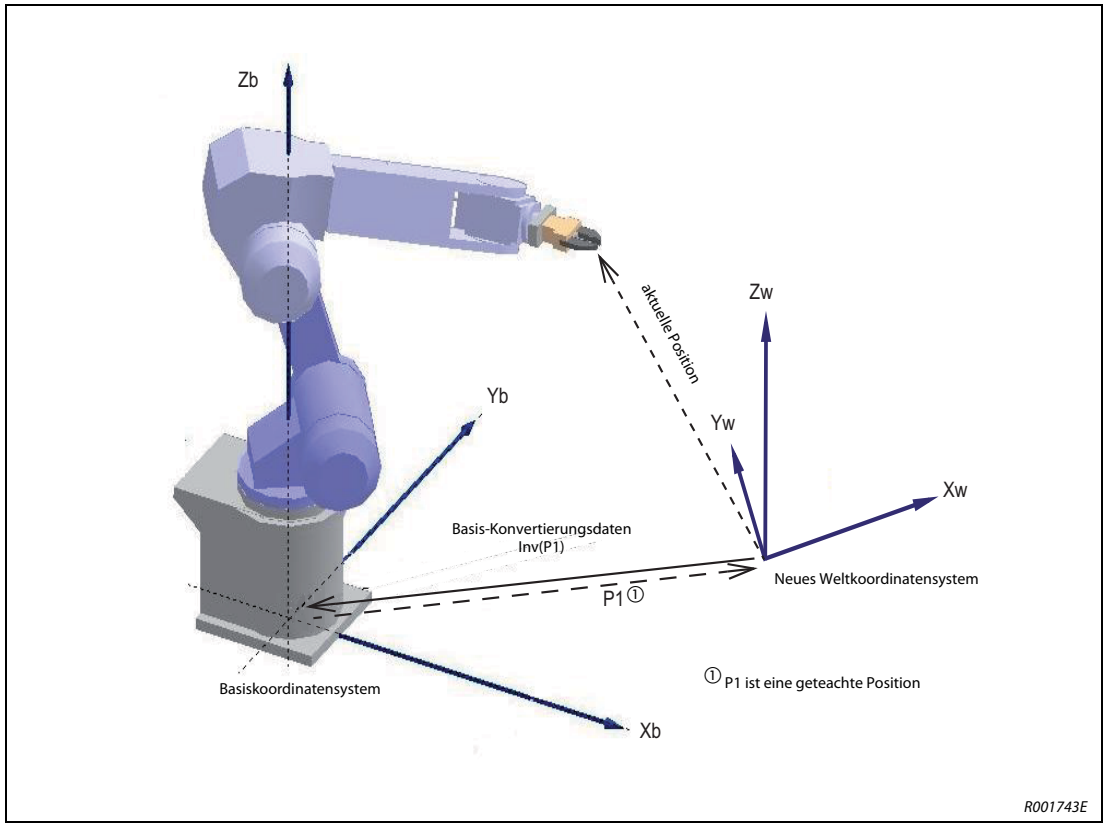


Abb. 5-11: *Basis-Konvertierungsdaten*

GEFAHR:

Da sich durch die Basis-Konvertierung der Bezugspunkt der aktuellen Position verändert, sind die bis dahin geteachten Daten nicht weiterverwendbar. Beim versehentlichen Anfahren von Daten, die vor der Konvertierung geteacht worden sind, kann sich der Roboter zu unvorhersehbaren Positionen bewegen. Dabei können Sach- und Personenschäden auftreten.

Prüfen Sie vor der Ausführung der Funktion die Beziehung zwischen den ursprünglich geteachten Positionen und den Positionen, die nach der Konvertierung angefahren werden, um einen einwandfreien Betrieb des Roboters zu gewährleisten.

5.2.3 Aufbau der Positionsdaten

Die Positionsdaten des Roboters bestehen aus sechs Elementen, die die Position (X, Y, Z) und Lage (A, B, C) der Handspitze (des Handflansches, wenn keine Werkzeugdaten eingestellt sind) im Raum beschreiben, und einem Stellungsmerker.
Jedes Element stellt einen Referenzwert im Weltkoordinatensystem des Roboters dar.

- X, Y, Z sind die Daten der Koordinaten. Sie geben die Position der Handspitze (TCP = Tool Center Point) des Roboters im kartesischen Koordinatensystem wieder. Sie werden in mm angegeben.
- A, B, C sind Orientierungsdaten der Roboterhand.
  A: Drehwinkel der X-Achse
  B: Drehwinkel der Y-Achse
  C: Drehwinkel der Z-Achse

Das folgende Diagramm zeigt den Bezug der Orientierungsdaten zu den Achsen. Es sind keine Basis-Konvertierungsdaten vorgegeben, so dass das Weltkoordinatensystem deckungsgleich mit dem Basiskoordinatensystem ist.

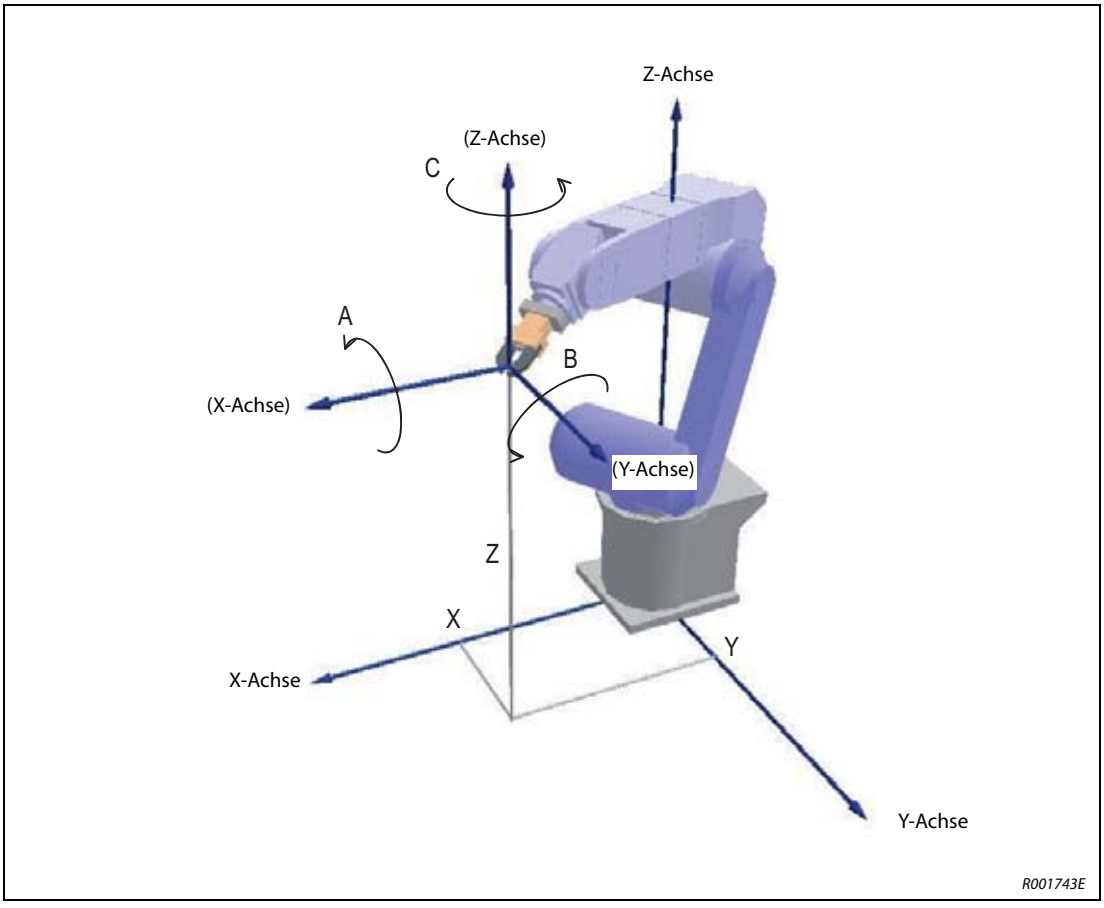


Abb. 5-12: *Bezug der Orientierungsdaten*

Die Orientierungsdaten A, B und C geben die Lage der Handspitze (des Handflansches, wenn keine Werkzeugdaten eingestellt sind) im Raum wieder. Sie beschreiben jeweils einen Drehwinkel um die Achsen X, Y und Z des Weltkoordinatensystems. Die Drehrichtung einer Schraube mit Rechtsgewinde, die sich ergibt, wenn die Schraube entlang den Achsen in Richtung steigender Werte geschraubt wird, ist die positive Drehrichtung. Bei der Berechnung der Winkel einer Position ist auf die Reihenfolge der Berechnung zu achten. Zunächst wird die Drehung um die Z-Achse (= C) durchgeführt, gefolgt von B und A. Die Reihenfolge ist für die letztendliche Positionierung von hoher Bedeutung, da bei einer unterschiedlichen Reihenfolge das Ergebnis (= die Position) komplett woanders liegt.

5.2.4 Werkzeugkoordinatensystem (Handflanschkoordinatensystem)

Um den zu steuernden Punkt vom Mittelpunkt des Handflansches in die Handspitze zu verlegen, müssen die Werkzeugdaten eingestellt werden. Die Werkzeugdaten beschreiben die Lage der Handspitze in Relation zum Mittelpunkt des Handflansches. Daher wird hier in erster Linie das Handflanschkoordinatensystem behandelt.

Die Erläuterungen zum Handflanschkoordinatensystem beziehen sich alle auf einen 6-achsigen Knickarmroboter. Eine Beschreibung für andere Modelle (5-achsiger Knickarmroboter und SCARA-Roboter) finden Sie im Abschn. 9.7 „Standard-Werkzeugkoordinaten“.

Handflanschkoordinatensystem

Das Koordinatensystem, dessen Ursprung im Mittelpunkt des Handflansches liegt, wird Handflanschkoordinatensystem genannt (siehe Abb. 5-13). Die Achsen werden mit Xm, Ym und Zm bezeichnet.

Die Zm-Achse verläuft durch den Mittelpunkt des Handflansches und steht senkrecht auf dessen Montageoberfläche. Die von der Montageoberfläche wegzeigende Richtung ist die Plusrichtung. Die Achsen Xm und Ym liegen in einer Ebene mit der Montageoberfläche. Die Achse Xm verläuft durch den Mittelpunkt der Montageoberfläche und die Bohrung für die Kalibriereinrichtung. Die Plusrichtung verläuft vom Mittelpunkt aus gesehen in die Richtung, die der Bohrung für die Kalibriereinrichtung gegenüberliegt.

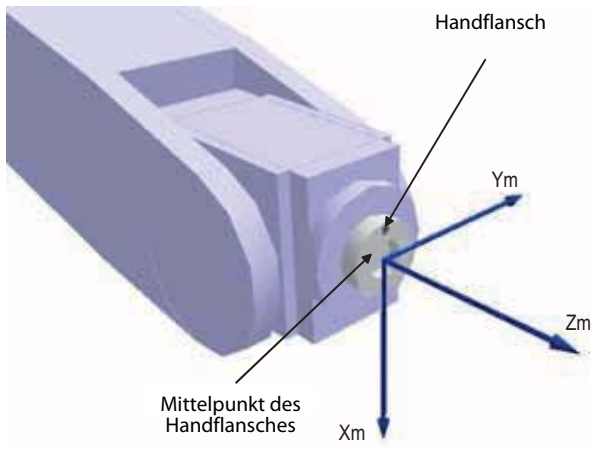


Abb. 5-13: *Das Handflanschkoordinatensystem*

Rotiert der Handflansch, so rotiert auch das Handflanschkoordinatensystem.

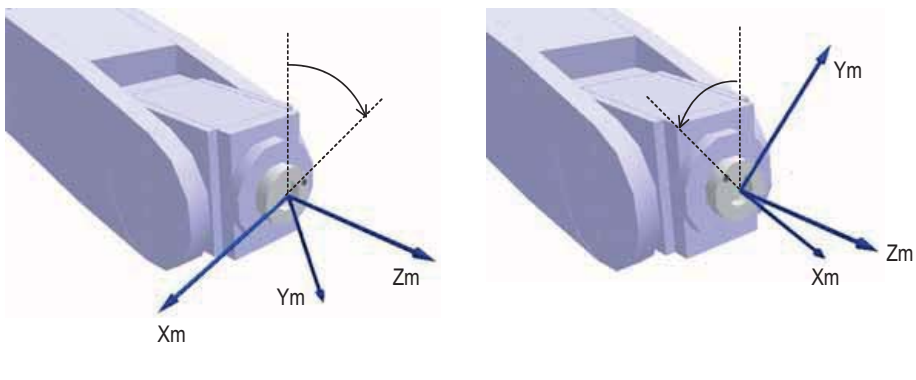


Abb. 5-14: *Rotation des Handflansches und des Handflanschkoordinatensystems*

Werkzeugkoordinatensystem

Das Werkzeugkoordinatensystem liegt mit seinem Ursprung im TCP der Hand. Es entsteht durch Verschiebung des Nullpunkt des Handflanschkoordinatensystems in den TCP. Die Achsen werden mit Xt, Yt und Zt bezeichnet.

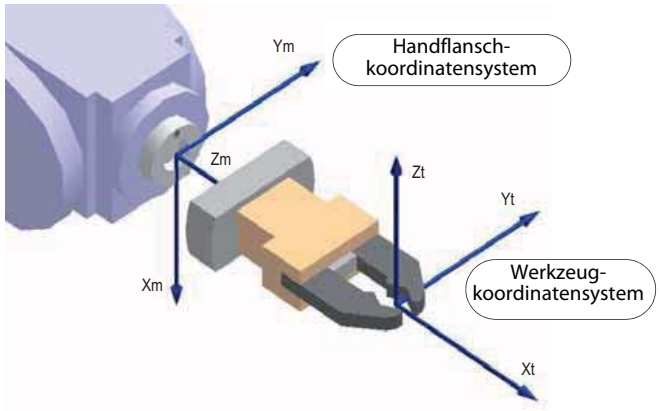


Abb. 5-15: *Das Werkzeugkoordinatensystem*

Die Werkzeugdaten bestehen aus den gleichen Elementen wie Positionsdaten.

- X, Y, Z sind die Abstände zum Handflanschkoordinatensystem. Der Nullpunkt des Handflanschkoordinatensystems müsste um diese Beträge verschoben werden, um deckungsgleich mit dem Werkzeugkoordinatensystem zu sein. Sie werden in mm angegeben.
- A, B, C sind Orientierungsdaten der Roboterhand.
  A: Drehwinkel der X-Achse
  B: Drehwinkel der Y-Achse
  C: Drehwinkel der Z-Achse

Verwendung des Werkzeugkoordinatensystems

- JOG-Betrieb und Teachen von Positionen
  Im Werkzeug-JOG-Betrieb wird die Vorderseite der Hand gesteuert. Dies vereinfacht eine Einstellung der Orientierung der Hand in Bezug auf das Werkstück oder die Orientierung eines Werkstücks, das von der Hand gehalten wird.

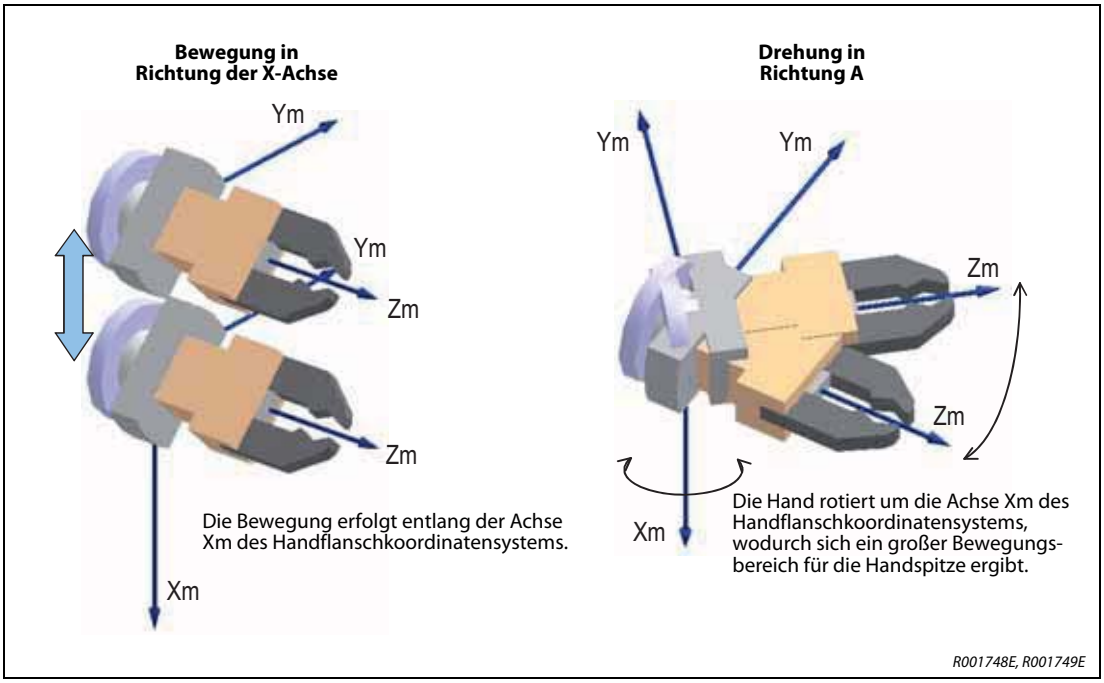


Abb. 5-16: *Werkzeug-JOG-Betrieb ohne eingestellte Werkzeugdaten*

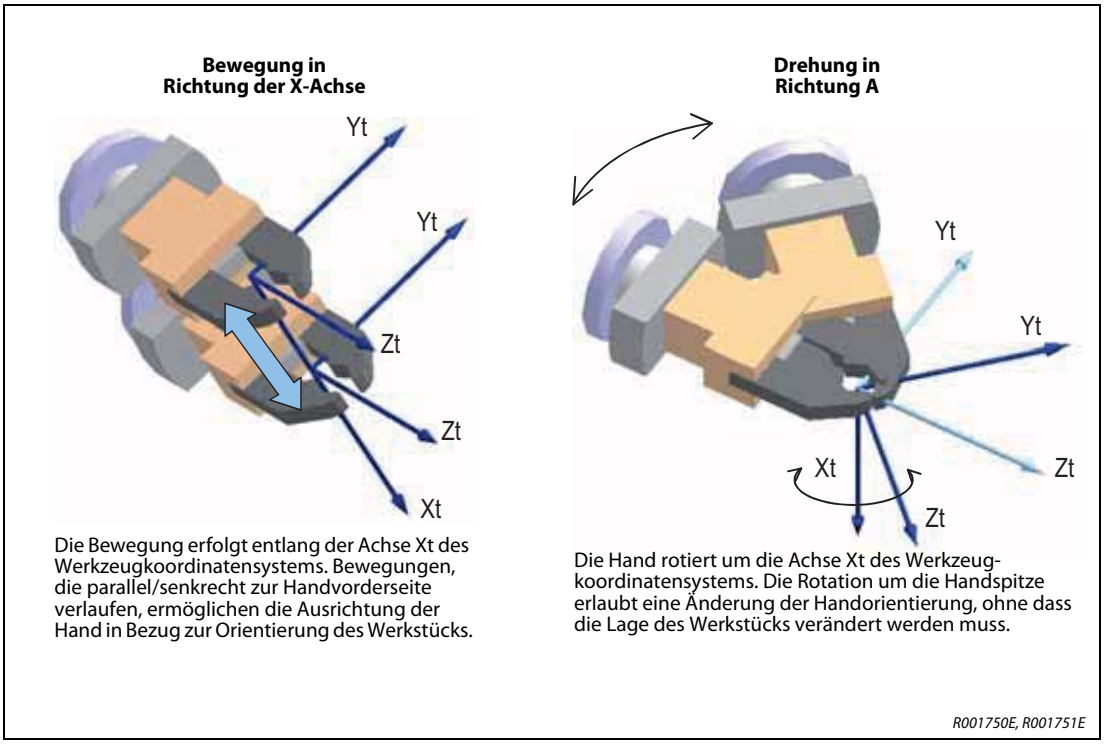

Bewegung in Richtung der X-Achse
Yt
Yt
Zt
Zt
Xt
Die Bewegung erfolgt entlang der Achse Xt des Werkzeugkoordinatensystems. Bewegungen, die parallel/senkrecht zur Handvorderseite verlaufen, ermöglichen die Ausrichtung der Hand in Bezug zur Orientierung des Werkstücks.
Drehung in Richtung A
Yt
Yt
Xt
Zt
Zt
Die Hand rotiert um die Achse Xt des Werkzeugkoordinatensystems. Die Rotation um die Handspitze erlaubt eine Änderung der Handorientierung, ohne dass die Lage des Werkstücks verändert werden muss.
R001750E, R001751E

Abb. 5-17: *Werkzeug-JOG-Betrieb mit eingestellten Werkzeugdaten*

- Automatikbetrieb
  Die Vorgabe von Abständen beim Aufnehmen oder Ablegen von Werkstücken, ermöglicht eine Roboterbewegung, während ein weiterzuberarbeitendes Werkstück entfernt oder zum nächsten Bearbeitungsschritt transportiert wird. Die Abstände können in Richtung der Z-Achse des Werkzeugkoordinatensystems vorgegeben werden.
  Um die Roboterhand z.B. zu einem Punkt zu bewegen, der 50 mm über der Ablageposition des Werkstücks liegt (siehe Abb. 5-18), verwenden Sie den Befehl: Mov P1, 50.
  Der Befehl bewirkt, dass sich der Roboter +50 mm entlang der Z-Achse im Werkzeugkoordinatensystem vom Punkt P1 wegbewegt.
  Die Vorgabe der Richtung der Z-Achse im Werkzeugkoordinatensystem erlaubt eine Anpassung des Roboters an die Orientierung des Werkstücks oder an die Betriebsbedingungen, was zu einer besseren Handhabbarkeit des Werkstücks führt.
  Im Beispiel in Abb. 5-18 ist die Roboterhand so ausgerichtet, dass sie das Werkstück einsetzen oder entfernen kann. Deshalb ist die Richtung der Z-Achse im Werkzeugkoordinatensystem so gewählt, dass sie mit der Orientierung des Werkstücks übereinstimmt.

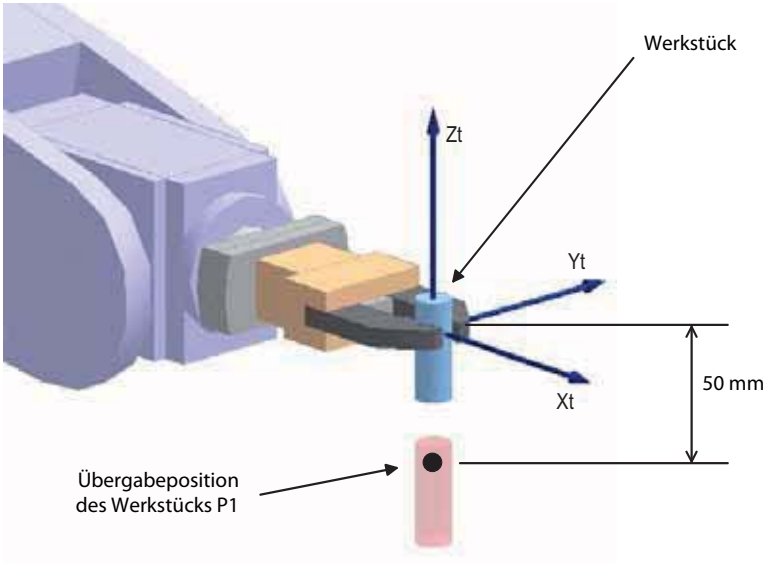


Abb. 5-18: *Bewegung zur Aufnahmeposition*

Die Einstellung der Werkzeugdaten ist dann sinnvoll, wenn die Orientierung des Werkstücks sich ändert oder wenn Werkstücke ausgerichtet werden müssen.

Beispiel▽ Mit dem folgenden Befehl können Werkstücke durch eine Drehung um ihre Mittelachse ausgerichtet werden (siehe Abb. 5-19):
Mov P1*(0,0,0,0,0,45)

Die Operation „*(0,0,0,0,0,45)“ bewirkt, dass C, aus den Elementen (X, Y, Z, A, B, C), um 45° gedreht wird. C beschreibt eine Rotation um die Z-Achse. Der Roboter führt also im Punkt P1 eine Drehung von 45° um die Achse Z (Zt im Werkzeugkoordinatensystem) aus.

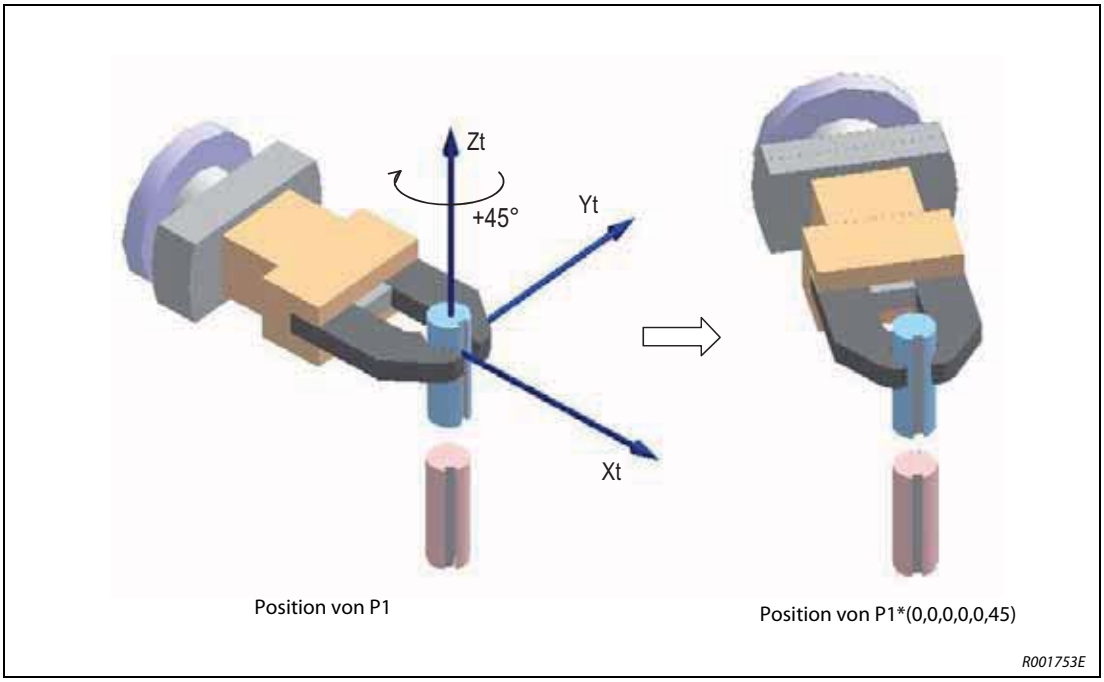


Abb. 5-19: *Drehung eines Werkstücks im Werkzeugkoordinatensystem*

△

6 MELFA-BASIC-V-Befehle

6.1 Allgemeine Hinweise

In den nachfolgenden Abschnitten finden Sie eine Auflistung aller MELFA-BASIC-V-Befehle und deren Anwendungsmöglichkeiten.

6.1.1 Beschreibung des verwendeten Formats

Funktion

Hier finden Sie eine Funktionsbeschreibung des Befehls.

Eingabeformat

Hier finden Sie das genaue Format zur Eingabe des Befehls. Befehlsparameter werden in spitzen Klammern „<>" angegeben. Die eckige Klammer „[ ]" kennzeichnet die wahlfreien Befehlsparameter. Die notwendige Eingabe eines Leerzeichens wird durch „□" dargestellt.

Programmbeispiel

Hier finden Sie die Verwendung des Befehls in einem Beispielprogramm.

Erläuterung

Hier finden Sie eine detaillierte Beschreibung, Besonderheiten usw. des Befehls.

6.2 Übersicht der MELFA-BASIC-V-Befehle

6.2.1 Alphabetische Übersicht

Befehl		Funktion	Seite
Accel	(Accelerate)	Beschleunigung und Verzögerung einstellen	6-8
Act	(Act)	Interrupt freigeben/sperren	6-10
Base	(Base)	Basis	6-12
CallP	(Call P)	Programm aufrufen	6-16
ChrSrch	(Character Search)	Zeichenkette in einer Zeichenketten-Feldvariablen suchen	6-18
Close	(Close)	Datei oder Kommunikationsleitung schließen	6-19
Clr	(Clear)	Löschen	6-20
Cmp Jnt	(Compliance Joint)	Achsenweichheit im Gelenkkoordinatensystem aktivieren	6-21
Cmp Pos	(Compliance Posture)	Achsenweichheit im XYZ-Koordinatensystem aktivieren	6-23
Cmp Tool	(Compliance Tool)	Achsenweichheit im Werkzeugkoordinatensystem aktivieren	6-26
Cmp Off	(Compliance OFF)	Achsenweichheit deaktivieren	6-29
CmpG	(Compliance Gain)	Achsenweichheit einstellen	6-30
Cnt	(Continous)	Roboterbewegung überschleifen	6-32
ColChk	(Col Check)	Kollisionsüberwachung aktivieren	6-36
ColLvl	(Col Level)	Empfindlichkeit der Kollisionsüberwachung einstellen	6-40
Com Off	(Communication OFF)	Kommunikations-Interrupt sperren	6-42
Com On	(Communication ON)	Kommunikations-Interrupt freigeben	6-43
Com Stop	(Communication Stop)	Kommunikations-Interrupt stoppen	6-44
Def Act	(Define Act)	Interrupt-Prozess definieren	6-45
Def Arch	(Define Arch)	Bogen definieren	6-48
Def Char	(Define Character)	Zeichenkettenvariable definieren	6-50
Def Fn	(Define function)	Funktion definieren	6-51
Def Inte/ Long/ Float/ Double	(Define Integer/Long/Float/ Double)	Numerische Variable definieren	6-53
Def IO	(Define IO)	Ein-/Ausgangsvariable definieren	6-55
Def Jnt	(Define Joint)	Gelenkvariable definieren	6-57
Def Plt	(Define pallet)	Palette definieren	6-58
Def Pos	(Define Position)	Positionsvariable definieren	6-61
Dim	(Dim)	Dimension einer Feldvariablen definieren	6-62
Dly	(Delay)	Verzögerung einstellen	6-64
End	(End)	Programmende	6-66
Error	(Error)	Fehler generieren	6-67
Fine	(Fine)	Feinpositionierung	6-68
Fine J	(Fine Joint)	Feinpositionierung bei Gelenk-Interpolation	6-70
Fine P	(Fine Pause)	Feinpositionierung über geradlinigen Abstand	6-72
For-Next	(For-next)	Programmschleife	6-74
FPrm	(FPRM)	Parameter definieren	6-76
GetM	(Get Mechanism)	Mechanismus definieren	6-77

Tab. 6-1: *Übersicht der Befehle (1)*

Befehl		Funktion	Seite
GoSub	(Go Subroutine)	Sprung zu einem Unterprogramm	6-79
GoTo	(Go To)	Sprung zu einer Marke	6-80
Hlt	(Halt)	Programmablauf stoppen	6-81
HOpen/ HClose	(Hand open/Hand close)	Handgreiferzustand festlegen	6-82
If Then Else	(If Then Else)	WENN ... DANN ... SONST-Schleife	6-83
Input #	(Input)	Daten einlesen	6-85
JOvrd	(J override)	Übersteuerung Gelenk-Interpolation	6-86
JRC	(Joint Roll Change)	Gelenkposition verändern	6-87
Label	(Label)	Sprungmarke	6-90
LoadSet	(Load set)	Hand- und Werkstückbedingung einstellen	6-91
Mov	(Move)	Bewegung mit Gelenk-Interpolation	6-93
Mva	(Move Arch)	Bewegung mit Bogen-Interpolation	6-95
Mvc	(Move C)	Kreis-Interpolation	6-97
Mvr	(Move R)	Kreis-Interpolation	6-99
Mvr2	(Move R2)	Kreis-Interpolation	6-101
Mvr3	(Move R3)	Kreis-Interpolation	6-103
Mvs	(Move S)	Bewegung mit Linear-Interpolation	6-105
MvTune	(Move Tune)	Bewegungsmodus	6-108
Mxt	(Move External)	Externe Steuerung	6-111
Oadl	(Optimum Acceleration/ Deceleration)	Optimale Beschleunigung/Abbremsung	6-113
On Com GoSub	(ON Communication Go Sub-routine)	Sprung zu einem Unterprogramm	6-115
On GoSub	(ON Go Subroutine)	Sprung zu einem Unterprogramm	6-117
On GoTo	(On Go To)	Programmverzweigung	6-119
Open	(Open)	Datei oder Kommunikationsleitung öffnen	6-121
Ovrd	(Override)	Übersteuerung	6-123
Plt	(Pallet)	Koordinaten für Palette berechnen	6-125
Prec	(Precision)	Verfahrweggenauigkeit erhöhen	6-128
Print #	(Print)	Daten übertragen	6-129
Priority	(Priority)	Priorität festlegen	6-131
RelM	(Release mechanism)	Mechanismuszuordnung aufheben	6-132
Rem	(Remarks)	Kommentar	6-133
Reset Err	(Reset Error)	Fehler zurücksetzen	6-134
Return	(Return)	Rücksprung zum Hauptprogramm Schritt hinter GoSub	6-135
Select Case	(Select case)	Prozess ausführen	6-137
Servo	(Servo)	Servo ein-/ausschalten	6-139
Skip	(Skip)	Sprung zum nächsten Programmschritt	6-140
Spd	(Speed)	Geschwindigkeit festlegen	6-141
SpdOpt	(Speed Optimize)	Geschwindigkeit optimieren	6-143
Title	(Title)	Programmtitel festlegen	6-146
Tool	(Tool)	Werkzeug-Konvertierungsdaten	6-147
Torq	(Torque)	Drehmomentgrenze definieren	6-149
Wait	(Wait)	Wartestatus definieren	6-151

Tab. 6-1: *Übersicht der Befehle (2)*

Befehl		Funktion	Seite
While~ WEnd	While End	Programmschleife	6-152
Wth	(With)	Anweisung hinzufügen	6-153
WthIf	(With If)	Anweisung hinzufügen, wenn ...	6-154
XClr	(X Clear)	Programmplatzauswahl zurücksetzen	6-155
XLoad	(X Load)	Programm laden	6-156
XRst	(X Reset)	Programm zurücksetzen	6-157
XRun	(X Run)	Programm starten	6-158
XStp	(X Stop)	Programm stoppen	6-160
Substitute	(Substitute)	Daten ersetzen	6-161

Tab. 6-1: *Übersicht der Befehle (3)*

6.2.2 Anwendungsspezifische Übersicht

In folgender Tabelle sind die MELFA-BASIC-V-Befehle in anwendungsspezifische Gruppen zusammengefasst. Die Reihenfolge wird durch die Verwendungshäufigkeit der Progammierbefehle bestimmt.

Anwendung	Befehl		Funktion	Seite
Befehle zur Bewegungssteuerung	Mov	(Move)	Bewegung mit Gelenk-Interpolation	6-93
	Mvs	(Move S)	Bewegung mit Linear-Interpolation	6-105
	Mvr	(Move R)	Kreis-Interpolation	6-99
	Mvr2	(Move R2)	Kreis-Interpolation	6-101
	Mvr3	(Move R3)	Kreis-Interpolation	6-103
	Mvc	(Move C)	Kreis-Interpolation	6-97
	Mva	(Move Arch)	Bewegung mit Bogen-Interpolation	6-95
	MvTune	(Move Tune)	Bewegungsmodus	6-108
	Ovrd	(Override)	Übersteuerung	6-123
	Spd	(Speed)	Geschwindigkeit festlegen	6-141
	JOvrd	(J override)	Übersteuerung Gelenk-Interpolation	6-86
	Cnt	(Continous)	Roboterbewegung überschleifen	6-32
	Accel	(Accelerate)	Beschleunigung und Verzögerung einstellen	6-8
	Cmp Jnt	(Compliance Joint)	Achsenweichheit im Gelenkkoordinatensystem aktivieren	6-21
	Cmp Pos	(Compliance Posture)	Achsenweichheit im XYZ-Koordinatensystem aktivieren	6-23
	Cmp Tool	(Compliance Tool)	Achsenweichheit im Werkzeugkoordinatensystem aktivieren	6-26
	Cmp Off	(Compliance OFF)	Achsenweichheit deaktivieren	6-29
	CmpG	(Compliance Gain)	Achsenweichheit einstellen	6-30
	Oadl	(Optimum Acceleration/ Deceleration)	Optimale Beschleunigung/Abbremsung	6-113
	Mxt	(Move External)	Externe Steuerung	6-111
	LoadSet	(Load set)	Hand- und Werkstückbedingung einstellen	6-91
	Prec	(Precision)	Verfahrweggenauigkeit erhöhen	6-128
	Torq	(Torque)	Drehmomentgrenze definieren	6-149
	JRC	(Joint Roll Change)	Gelenkposition verändern	6-87
	Fine	(Fine)	Feinpositionierung	6-68
	Fine J	(Fine Joint)	Feinpositionierung bei Gelenk-Interpolation	6-70
	Fine P	(Fine Pause)	Feinpositionierung über geradlinigen Abstand	6-72
	Servo	(Servo)	Servo ein-/ausschalten	6-139
	Wth	(With)	Anweisung hinzufügen	6-153
	Wthlf	(With If)	Anweisung hinzufügen, wenn ...	6-154

Tab. 6-2: *Einteilung der Befehle in anwendungsspezifische Gruppen (1)*

Anwendung	Befehl		Funktion	Seite
Befehle zur Programm-steuerung	Rem	(Remarks)	Kommentar	6-133
	If Then Else EndIf	(If Then Else)	Bedingte Verzweigung	6-83
	Select Case	(Select case)	Prozess ausführen	6-137
	GoTo	(Go To)	Sprung zu einer Marke	6-80
	GoSub (Return)	(Go Subroutine)	Sprung zu einem Unterprogramm	6-79
	Reset Err	(Reset Error)	Fehler zurücksetzen	6-134
	CallP	(Call P)	Programm aufrufen	6-16
	FPrm	(FPRM)	Parameter definieren	6-76
	Dly	(Delay)	Verzögerung einstellen	6-64
	Hlt	(Halt)	Programmablauf stoppen	6-81
	End	(End)	Programmende	6-66
	On GoSub	(ON Go Subroutine)	Sprung zu einem Unterprogramm	6-117
	On GoTo	(On Go To)	Programmverzweigung	6-119
	For-Next	(For-next)	Programmschleife	6-74
	While~ WEnd	While End	Programmschleife	6-152
	Open	(Open)	Datei oder Kommunikationsleitung öffnen	6-121
	Print #	(Print)	Daten übertragen	6-129
	Input #	(Input)	Daten einlesen	6-85
	Close	(Close)	Datei oder Kommunikationsleitung schließen	6-19
	ColChk	(Col Check)	Kollisionsüberwachung aktivieren	6-36
	On Com GoSub	(ON Communication Go Sub-routine)	Sprung zu einem Unterprogramm	6-115
	Com On	(Communication ON)	Kommunikations-Interrupt freigeben	6-43
	Com Off	(Communication OFF)	Kommunikations-Interrupt sperren	6-42
	Com Stop	(Communication Stop)	Kommunikations-Interrupt stoppen	6-44
	HOpen/ HClose	(Hand open/Hand close)	Handgreiferzustand festlegen	6-82
	Error	(Error)	Fehler generieren	6-67
	Skip	(Skip)	Sprung zum nächsten Programmschritt	6-140
	Wait	(Wait)	Wartestatus definieren	6-151
	Clr	(Clear)	Löschen	6-20

Tab. 6-2: *Einteilung der Befehle in anwendungsspezifische Gruppen (2)*

Anwendung	Befehl		Funktion	Seite
Definitions-befehle	Dim	(Dim)	Dimension einer Feldvariablen definieren	6-62
	Def Plt	(Define pallet)	Palette definieren	6-58
	Plt	(Pallet)	Koordinaten für Palette berechnen	6-125
	Def Act	(Define Act)	Interrupt-Prozess definieren	6-45
	Act	(Act)	Interrupt freigeben/sperren	6-10
	Def Arch	(Define Arch)	Bogen definieren	6-48
	Def Jnt	(Define Joint)	Gelenkvariable definieren	6-57
	Def Pos	(Define Position)	Positionsvariable definieren	6-61
	Def Inte/ Long/ Float/ Double	(Define Integer/Long/Float/ Double)	Numerische Variable definieren	6-53
	Def Char	(Define Character)	Zeichenkettenvariable definieren	6-50
	Def IO	(Define IO)	Ein-/Ausgangsvariable definieren	6-55
	Def Fn	(Define function)	Funktion definieren	6-51
	Tool	(Tool)	Werkzeug-Konvertierungsdaten	6-147
	Base	(Base)	Basis	6-12
Multitasking-Befehle	XLoad	(X Load)	Programm laden	6-156
	XRun	(X Run)	Programm starten	6-158
	XStp	(X Stop)	Programm stoppen	6-160
	XRst	(X Reset)	Programm zurücksetzen	6-157
	XClr	(X Clear)	Programmplatzauswahl zurücksetzen	6-155
	GetM	(Get Mechanism)	Mechanismus definieren	6-77
	RelM	(Release mechanism)	Mechanismuszuordnung aufheben	6-132
	Priority	(Priority)	Priorität festlegen	6-131
	Reset Err	(Reset Error)	Fehler zurücksetzen	6-134
Andere Befehle	ChrSrch	(Character Search)	Zeichenkette in einer Zeichenketten-Feldvariablen suchen	6-18
	Get Pos	(Get Position)	Reserviert	—

Tab. 6-2: *Einteilung der Befehle in anwendungsspezifische Gruppen (3)*

6.3 Detaillierte Befehlsbeschreibung

In diesem Abschnitt finden Sie eine detaillierte Beschreibung sowie Programmbeispiele zur Anwendung der Befehle.

6.3.1 Accel (Accelerate)

Funktion: Beschleunigung und Abbremsung einstellen

Legt den Wert für die Beschleunigung und Abbremsung in Prozent fest. Der Wert ist auch während der optimalen Beschleunigung/Abbremsung wirksam.

Ist die optimale Beschleunigung/Abbremsung über die Anweisung Oadl aktiviert, ergibt sich die Beschleunigungs-/Bremszeit unter Berücksichtigung des Wertes, der in der Variablen M_SetAdl festgelegt ist.

Eingabeformat

```
Accel □ [<Beschleunigung>][,<Abbremsung>],
        [<Beschleunigung bei Aufwärtsbewegung>],
        [<Abbremsung bei Aufwärtsbewegung>],
        [<Beschleunigung bei Abwärtsbewegung>],
        [<Abbremsung bei Abwärtsbewegung>]
```

<Beschleunigung/Abbremsung>	Legt den Prozentwert der Beschleunigung/ Abbremsung in einem Einstellbereich von 1 bis 100 % vom Stillstand bis zur maximalen Geschwindigkeit fest Die Werte können als Konstante oder Variable angegeben werden. Erfolgt keine Angabe, werden die Werte auf 100 [%] gesetzt. 100 % entsprechen dabei der maximalen Beschleunigung bzw. Abbremsung.
<Beschleunigung/Abbremsung bei Aufwärtsbewegung>	Legt die Beschleunigung/Abbremsung bei einer Aufwärtsbewegung mit Bogen-Interpolation (Mva-Befehl) fest Die Werte können als Konstante oder Variable angegeben werden. Erfolgt keine Angabe, werden die Werte auf 100 [%] gesetzt.
<Beschleunigung/Abbremsung bei Abwärtsbewegung>	Legt die Beschleunigung/Abbremsung bei einer Abwärtsbewegung mit Bogen-Interpolation (Mva-Befehl) fest Die Werte können als Konstante oder Variable angegeben werden. Erfolgt keine Angabe, werden die Werte auf 100 [%] gesetzt.

Programmbeispiel

```
1 Accel 50,100          'Beschleunigung/Abbremsung für große Last
                        einstellen (Bei einem Grundwert der
                        Beschleunigungs-/Abbremszeit von 0,2 s ist hier im
                        Beispiel eine Beschleunigungszeit von 0,4 s und eine
                        Abbremszeit von 0,2 s wirksam.)
2 Mov P1                'Position P1 mittels Gelenk-Interpolation anfahren
3 Accel 100,100         'Beschleunigung/Abbremsung für Standardlast
                        einstellen
4 Mov P2                'Position P2 mittels Gelenk-Interpolation anfahren
5 Def Arch 1,10,10,25,25,1,0,0  'Bogen definieren
6 Accel 100,100,20,20,20,20     'Beschleunigung/Abbremsung für die Bogen-
                                Interpolation für Auf- und Abwärtsbewegung auf
                                20 % reduzieren
7 Mva P3,1              'Position P3 über Bogen 1 mittels Bogen-
                        Interpolation anfahren
```

Erläuterung

- Die maximale Beschleunigung/Abbremsung ist vom verwendeten Robotertyp abhängig. Stellen Sie die Beschleunigung/Abbremsung als Prozentwert des Maximalwertes ein. Als Standardwerte sind die Werte 100, 100 eingestellt.
- Die Beschleunigungs- bzw. Abbremszeit berechnet sich aus:

$$\text{Zeit} = \frac{100\ \%}{\text{Beschleunigung/Abbremsung [\%]}} \times 0{,}2\ \text{s}$$

- Einstellungen größer 100 werden bei einigen Robotermodellen automatisch auf 100 gesetzt. Eine Einstellung größer 100 kann die Lebensdauer des Roboters verkürzen und die Wahrscheinlichkeit von Fehlern aufgrund von Geschwindigkeits- oder Lastüberschreitungen steigt. Vermeiden Sie daher Einstellungen größer 100.
- Die über diesen Befehl eingestellte Beschleunigung/Abbremsung wird beim Zurücksetzen des Progamms und bei Ausführung der End-Anweisung auf die Standardwerte zurückgesetzt.
- Die Verfahrkurve für eine kontinuierliche, gleichmäßige Bewegung (Cnt freigegeben) kann von der Verfahrkurve mit Beschleunigung abweichen. Die Größe der Abweichung ist abhängig vom Wert der eingestellten Beschleunigungszeit. Für eine gleichmäßige Bewegung mit einer konstanten Geschwindigkeit sollten die Beschleunigungs- und die Abbremszeit gleich sein. In der Grundeinstellung ist die Cnt-Einstellung gesperrt.
- Die Einstellung des Accel-Befehls ist auch bei aktivierter optimaler Beschleunigung/Abbremsung (Oadl On) wirksam.

Steht in Beziehung zu folgenden Befehlen:

Mxt, LoadSet

Steht in Beziehung zu folgenden Systemvariablen:

M_Acl/M_DAcl/M_NAcl/M_NDAcl/M_AclSts, M_SetAdl

Steht in Beziehung zu folgenden Parametern:

JADL

6.3.2 Act (Act)

Funktion: Interrupt freigeben/sperren

Über diesen Befehl kann die Ausführung von Interrupt-Prozessen während des Betriebes freigegeben oder gesperrt werden.

Eingabeformat

Act □ [<Priorität> = <1/0>

<Priorität>	Gibt den Interrupt frei oder sperrt ihn 1 ≤ Priorität ≤ 8 Legt die mit der Anweisung Def Act definierte Priorität des Interrupts fest Hinter dem Act-Befehl muss ein Leerzeichen stehen. Die Schreibweise Act1 wird als Anweisung zur Deklaration einer Variablen gewertet.
<1/0>	1 = freigeben, 0 = sperren

Programmbeispiel

Beim Einschalten des Eingangssignals 1 während der Verfahrbewegung von P1 nach P2, wird eine Warteschleife durchlaufen, bis das Signal wieder auf „0“ gesetzt wird.

```
1  Def Act 1,M_In(1) = 1 GoSub *INTR   'Weist den Eingang 1 dem Interrupt 1 zu
2  Mov P1                              'Position P1 mittels Gelenk-Interpolation anfahren
3  Act 1 = 1                           'Interrupt 1 freigeben
4  Mov P2                              'Position P2 mittels Gelenk-Interpolation anfahren
5  Act 1 = 0                           'Interrupt 1 sperren
   :
10 *INTR                               'Ändert sich das Eingangssignal 1 auf EIN
                                       (1) während der Roboter sich von P1 nach
                                       P2 bewegt, wird die Warteschleife
                                       durchlaufen.
11 If M_In(1) = 1 GoTo *INTR           'Durchläuft die Warteschleife bis M_In(1)
12 Return 0                            auf „0“ gesetzt wird.
```

Beim Einschalten des Eingangssignals 1 während der Verfahrbewegung von P1 nach P2, wird der Betrieb unterbrochen und das Ausgangssignal 10 ausgeben.

```
1  Def Act 1,M_In(1) = 1 GoSub *INTR   'Weist den Eingang 1 dem Interrupt 1 zu
2  Mov P1                              'Position P1 mittels Gelenk-Interpolation anfahren
3  Act 1 = 1                           'Interrupt 1 freigeben
4  Mov P2                              'Position P2 mittels Gelenk-Interpolation anfahren
   :
10 *INTR                               'Ändert sich das Eingangssignal 1 auf EIN
                                       (1) während der Roboter sich von P1 nach
                                       P2 bewegt, wird der Betrieb unterbrochen.
11 Act 1 = 0                           'Interrupt 1 sperren
12 M_Out(10) = 1                       'Ausgabe des Ausgangssignals 10
13 Return 1                            'Springt einen Schritt hinter den Programmschritt,
                                       in dem der Interrupt aufgetreten ist
```

Erläuterung

- Beim Programmstart ist der Interrupt mit der Priorität 0 freigegeben. Wenn der Interrupt mit der Priorität 0 gesperrt ist, werden die Interrupts der Prioritäten 1 bis 8 nicht freigegeben, auch wenn sie auf „freigegeben" gesetzt sind.
- Die Interrupts der Prioritäten 1 bis 8 sind beim Programmstart gesperrt.
- Ein Interrupt kann nur ausgeführt werden, wenn folgende Bedingungen erfüllt sind:
  - Der Interrupt der Priorität 0 ist freigegeben.
  - Der Status der Def-Act-Anweisung ist definiert worden.
  - Der Interrupt, der in der Def-Act-Anweisung festgelegt wurde, ist durch die Act-Anweisung freigegeben.
- Ein Rücksprung aus einer Interruptroutine kann entweder durch Return 0 oder Return 1 erfolgen. Sperren Sie den Interrupt, wenn der Rücksprung über Return 1 zu dem Programmschritt, der dem Schritt mit dem Interruptaufruf folgt, erfolgte. Wird der Interrupt nicht gesperrt und die Interrupt-Bedingung ist erfüllt, erfolgt eine erneute Ausführung der Interrupt-Routine und der Schritt kann beim Rücksprung übersprungen werden.
- Auch wenn der Roboter sich in einer Interpolation befindet, wird ein mit Def Act definierter Interrupt ausgeführt.
- Während eines Interruptprozesses wird der entsprechende Interrupt auf gesperrt gesetzt.
- Ein Kommunikations-Interrupt hat eine höhere Priorität als ein mit Def Act definierter Interrupt.
- Die Reihenfolge der Prioritäten ist: COM > Act > WthIf (Wth).

Steht in Beziehung zu folgenden Befehlen:

Def Act, Return

6.3.3 Base (Base)

Funktion: Basis-Konvertierungsdaten

Dieser Befehl ermöglicht eine Verschiebung oder Drehung des Weltkoordinatensystems, das die Referenz zur Steuerung der aktuellen Roboterposition darstellt. Dazu stehen zwei Möglichkeiten zur Verfügung. Bei der ersten Möglichkeit erfolgt eine direkte Festlegung der Basis-Konvertierungsdaten, bei der zweiten Möglichkeit wird die Nummer eines vordefinierten Werkstückkoordinatensystems gewählt. Beachten Sie, dass diese Funktion großen Einfluss auf geteachte Daten und den JOG-Betrieb des Roboters hat. Lesen Sie die daher Beschreibungen im Abschn. 5.2 „Koordinatensysteme des Roboters“ aufmerksam durch und verwenden Sie die Funktion mit äußerster Sorgfalt.

Eingabeformat

Base ☐ <Basis-Konvertierungsdaten>
Base ☐ <Nummer des Basiskoordinatensystems>

<Basis-Konvertierungsdaten>

Legt die Basis-Konvertierungsdaten in Form von Positionskonstanten oder -variablen fest
Die Werte (Koordinatenwerte) sind Positionsdaten, die den Ursprung des Basiskoordinatensystems vom neu erzeugten Weltkoordinatensystem aus gesehen beschreiben.

<Nummer des Basiskoordinatensystems>

Die Werkseinstellung oder die im gewählten Parameter eingestellten Werte (Werkstückkoordinatensystem) werden als Basis-Konvertierungsdaten verwendet.
Der Wert ist eine Konstante in numerischer Form oder eine Variable von 0 bis 8.

0: P_NBASE (Werkseinstellung des System) wird verwendet.
1–8: Jeder Wert entspricht einem der Parameter (Werkstückkoordinatensysteme) WK1CORD–WK8CORD

Bei Verwendung einer Zahl vom Typ Real oder einer Zahl vom Typ Real mit doppelter Genauigkeit werden die Nachkommastellen abgerundet.

Programmbeispiel

Festlegung durch Basis-Konvertierungsdaten

```
1 Base (50,100,0,0,0,90)    'Mit Hilfe der Basis-Konvertierungsdaten in Form einer
                             Konstanten wird ein neues Weltkoordinatensystem
                             definiert
2 Mvs P1                    'Position P1wird im neuen Weltkoordinatensystem
                             mittels Linear-Interpolation anfahren
3 Base P2                   'Mit Hilfe der Basis-Konvertierungsdaten in Form einer
                             Konstanten wird ein neues Weltkoordinatensystem
                             definiert
4 Mvs P1                    'Position P1 wird im neuen Weltkoordinatensystem
                             mittels Linear-Interpolation anfahren
5 Base 0                    'Zurücksetzen des Weltkoordinatensystems auf den
                             Standardwert (das Weltkoordinatensystem wird auf
                             den Standardwert P_NBase gesetzt)
```

Festlegung durch die Nummer des Basiskoordinatensystems

```
1 Base 1                    'Das Werkstückkoordinatensystem 1
                             (Parameter: WK1CORD) wird als neues
                             Weltkoordinatensystem festgelegt
2 Mvs P1                    'Position P1wird im neuen Weltkoordinatensystem
                             mittels Linear-Interpolation anfahren
3 Base 2                    'Das Werkstückkoordinatensystem 2
                             (Parameter: WK2CORD) wird als neues
                             Weltkoordinatensystem festgelegt
4 Mvs P1                    'Position P1 wird im neuen Weltkoordinatensystem
                             mittels Linear-Interpolation anfahren
5 Base 0                    'Zurücksetzen des Weltkoordinatensystems auf den
                             Standardwert (das Weltkoordinatensystem wird auf
                             den Standardwert P_NBase gesetzt)
```

Erläuterung

- Die Werte für die Basis-Konvertierung (Koordinatenwerte) sind Positionsdaten, die den Ursprung des Basiskoordinatensystems vom dem Weltkoordinatensystem aus gesehen beschreiben, das durch die Konvertierung neu erzeugt wird.
  Möchten Sie also die aktuelle Position verwenden, um die Basis-Konvertierungsdaten mit Hilfe der Fram-Funktion o.Ä. festzulegen, so führen Sie eine inverse Konvertierung der Koordinatenwerte durch (z.B. Base Inv(P19)).
  Legen Sie das Werkstückkoordinatensystem über eine Nummer fest, müssen Sie die inverse Konvertierung nicht selbst durchführen, da diese intern ausgeführt wird
  Die X-, Y- und Z-Koordinaten geben die parallele Verschiebung des Basiskoordinatensystems in Bezug auf das Weltkoordinatensystem an. Die Komponenten A, B und C geben dabei die Drehwinkel des Basiskoordinatensystems in Bezug auf das Weltkoordinatensystem an.
  X = Paralleler Abstand zur X-Achse
  Y = Paralleler Abstand zur Y-Achse
  Z = Paralleler Abstand zur Z-Achse
  A = Drehwinkel um die X-Achse
  B = Drehwinkel um die Y-Achse
  C = Drehwinkel um die Z-Achse
  L1 = Weg der Zusatzachse 1
  L2 = Weg der Zusatzachse 2
- Aus Sicht des Koordinatenursprungs werden Winkel in Uhrzeigerrichtung positiv gewertet.
- Die Werte der Stellungsmerker sind ohne Bedeutung.
- Die Änderungen des mit dem Base-Befehl geänderten Basiskoordinatensystems werden im Parameter MEXBS gespeichert und bleiben auch nach Ausschalten der Spannungsversorgung erhalten.
- Die für den Base-Befehl zugelassen Achsen sind vom Robotermodell abhängig (siehe Abschn. 9.8 „Standard-Basiskoordinaten“).
  Weitere Informationen zu diesem Befehl finden Sie in Abschn. 5.2.1 „Beschreibung der Koordinatensysteme“ und Abschn. 5.2.2 „Basis-Konvertierung“.
- Der Standardwert ist P_NBase = (0,0,0,0,0,0)(0,0).

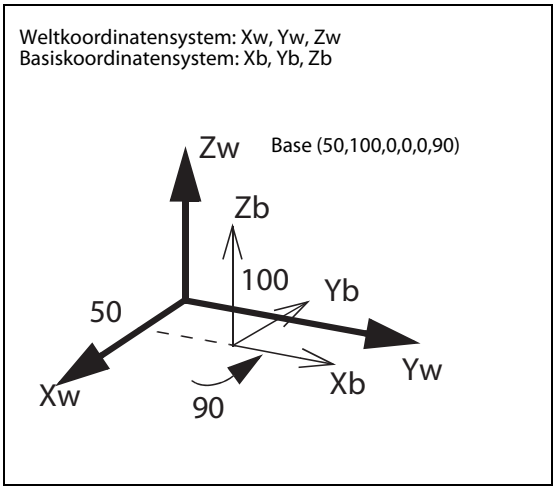


Abb. 6-1:
Basis-Konvertierungsdaten

R001755E

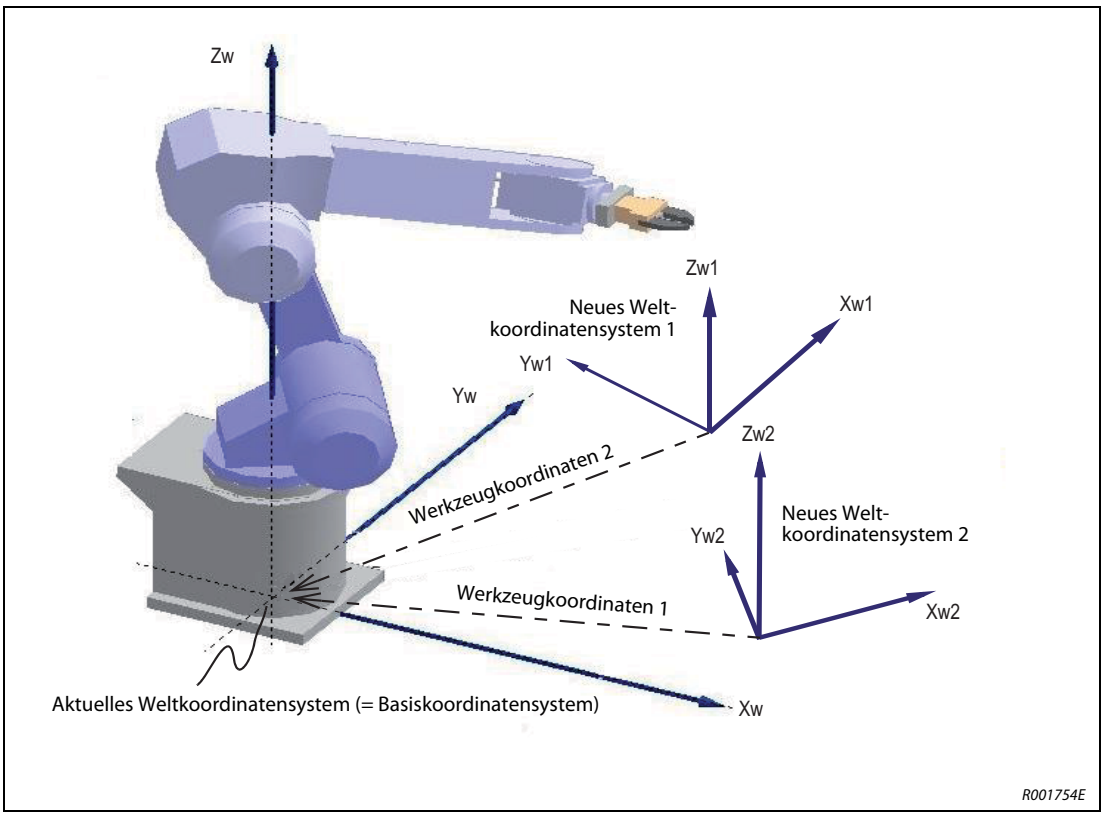


Abb. 6-2: *Basis-Konvertierung mit einem Werkstückkoordinatensystem, das über einen numerischen Wert festgelegt ist*

GEFAHR:

Da sich durch die Basis-Konvertierung der Bezugspunkt der aktuellen Position verändert, sind die bis dahin geteachten Daten nicht weiterverwendbar. Beim versehentlichen Anfahren von Daten, die vor der Konvertierung geteacht worden sind, kann sich der Roboter zu unvorhersehbaren Positionen bewegen. Dabei können Sach- und Personenschäden auftreten.

Prüfen Sie vor der Ausführung der Funktion die Beziehung zwischen den ursprünglich geteachten Positionen und den Positionen, die nach der Konvertierung angefahren werden, um einen einwandfreien Betrieb des Roboters zu gewährleisten.

Steht in Beziehung zu folgenden Parametern:

MEXBS, WKnCORD (n = 1 bis 8), MEXBSNO

Steht in Beziehung zu folgenden Systemvariablen:

M_BsNo, P_Base (aktuelle Basis-Konvertierungsdaten), P_NBase (Grundeinstellung), P_WkCord

6.3.4 CallP (Call P)

Funktion: Programm aufrufen

Führt das aufgerufene Programm aus (siehe auch GoSub-Befehl für Unterprogrammaufrufe). Der Rücksprung ins Hauptprogramm erfolgt bei Ausführung der End-Anweisung oder der letzten Zeile des Unterprogramms.

Eingabeformat

CallP □ "<Programmname>" [,<Parameter>[,<Parameter>] ...]

<Programmname>	Legt den Programmnamen als Zeichenkettenkonstante oder Zeichenkettenvariable fest Weitere Hinweise zur Vergabe von Programmnamen finden Sie in Abschn. 4.2.1.
<Parameter>	Legt die Variablen fest, die beim Aufruf des Programmes übergeben werden Es können maximal 16 Variablen übergeben werden.

Programmbeispiel

Übergabe der Variablen an das aufgerufene Programm

Hauptprogramm

```
1  M1 = 10                      'Weist M1 den Wert 10 zu
2  CallP "100",M1,P1,P2         'Aufruf des Programms 100 und Übergabe der
                                 Variablen M1, P1, P2
3  M1 = 1                       'Weist M1 den Wert 1 zu
4  CallP "100",M1,P1,P2         'Aufruf des Programms 100 und Übergabe der
                                 Variablen M1, P1, P2
   :
10 CALLP "10",M2,P3,P4          'Aufruf des Programms 10 und Übergabe der
                                 Variablen M2, P3, P4
   :
15 End                          'Programmende
```

Unterprogramm

```
1  FPrm M01,P01,P02             'Definiert die Variablen M01, P01, P02
2  If M01<> 0 Then GoTo *LBL1   'Sprung zur Marke LBL1, falls M01 ungleich 0 ist
3  Mov P01                      'Position mittels Gelenk-Interpolation anfahren
4  *LBL1                        'Legt die Sprungmarke LBL1 fest
5  Mvs P02                      'Position mittels Linear-Interpolation anfahren
6  End                          'Rücksprung in Hauptprogramm
```

Bei Ausführung der Schritte 2 und 4 des Hauptprogramms werden die Werte der Variablen M1, P1 und P2 in die Variablen M01, P01 und P02 des Unterprogramms übertragen. Bei Ausführung des Schritts 10 des Hauptprogramms werden die Werte der Variablen M2, P3 und P4 in die Variablen M01, P01 und P02 des Unterprogramms übertragen.

Keine Übergabe der Variablen an das aufgerufene Programm

Hauptprogramm

```
1  Mov P1                  'Position mittels Gelenk-Interpolation anfahren
2  CallP "20"              'Aufruf des Programms 20
3  Mov P2                  'Position mittels Gelenk-Interpolation anfahren
4  CallP "20"              'Aufruf des Programms 20
5  End                     'Programmende
```

Programm "20"

```
201  Mov P1                'Position P1 des Unterprogramms entspricht nicht
                           der Position P1 des Hauptprogramms
202  Mvs P002              'Position P002 mittels Linear-Interpolation anfahren
203  MOut(17) = 1          'Ausgang 17 auf „1“ setzen
204  End                   'Rücksprung in Hauptprogramm
```

Erläuterung

- Der Rücksprung ins Hauptprogramm aus einem über die CallP-Anweisung aufgerufenen Programm (Unterprogramm) erfolgt mit Ausführung der End-Anweisung (analog zur Return-Anweisung beim Befehl GoSub). Fehlt die End-Anweisung, erfolgt der Rücksprung ins Hauptprogramm nach Abarbeitung der letzten Unterprogrammzeile.
- Die Definition von Variablen zur Variablenübergabe erfolgt zu Beginn des Unterprogramms mit der FPrm-Anweisung.
- Weicht eine Variable in der CallP-Anweisung von der im Programmdefinierten Variablen (FPrm) ab, erfolgt eine Fehlermeldung.
- Weicht die Anzahl der Variablen in der CallP-Anweisung von der Anzahl der im Programm definierten Variablen ab, erfolgt eine Fehlermeldung.
- Wird das Programm zurückgesetzt, geht die Steuerung auf den Anfang des Hauptprogramms zurück.
- Das aufgerufene Programm hat keinen Einfluss auf die Anweisungen Def Act, Def Fn, Def Plt und Dim im aufrufenden Programm. Sobald das aufgerufene Programm zurückspringt, werden sie wieder gültig.
- Die Geschwindigkeits-, Werkzeugdaten (TCP) und die Oadl-Einstellung bleiben gültig, die über die Befehle Accel und Spd festgelegten Werte sind ungültig.
- Innerhalb eines Unterprogramms kann über die CallP-Anweisung ein weiteres Unterprogramm aufgerufen werden. Es ist nicht möglich das aufrufende oder ein in einem anderen Programmplatz aktives Programm aufzurufen. Ein Programm kann sich nicht selber aufrufen.
- Mit der CallP-Anweisung können bis zu 8 Programme von einem Programm aus aufgerufen werden.
- Variablenwerte können über Parameter vom aufrufenden Programm an das aufgerufene Programm übergeben werden. Die Ergebnisse, die im aufgerufenen Programm berechnet wurden, können nicht über die Parameter an das aufrufende Programm übergeben werden. Verwenden Sie externe Variablen, um die Werte zu übergeben.

Steht in Beziehung zu folgenden Befehlen:

FPrm

6.3.5 ChrSrch (Character search)

Funktion: Zeichenkette suchen

Sucht eine Zeichenkette innerhalb einer Feldvariablen.

Eingabeformat

ChrSrch □ <Zeichenketten-Feldvariable>,<Zeichenkette>, <Suchergebnis>

<Zeichenketten-Feldvariable>	Legt die zu durchsuchende Zeichenketten-Feldvariable fest
<Zeichenkette>	Legt die zu suchende Zeichenketten fest
<Suchergebnis>	Legt den Speicherort für die Nummer des gefundenen Feldelements fest

Programmbeispiel

```
1  DIM C1$(10)                    'Deklariert eine Feldvariable
2  C1$(1) = "ABCDEFG"             'Legt das erste Feldelement fest
3  C1$(2) = "MELFA"               'Legt das zweite Feldelement fest
4  C1$(3) = "BCDF"                'Legt das dritte Feldelement fest
5  C1$(4) = "ABD"                 'Legt das vierte Feldelement fest
6  C1$(5) = "XYZ"                 'Legt das fünfte Feldelement fest
7  C1$(6) = "MELFA"               'Legt das sechste Feldelement fest
8  C1$(7) = "CDF"                 'Legt das siebte Feldelement fest
9  C1$(8) = "ROBOT"               'Legt das achte Feldelement fest
10 C1$(9) = "FFF"                 'Legt das neunte Feldelement fest
11 C1$(10) = "BCD"                'Legt das zehnte Feldelement fest
12 CHRSRCH C1$(1), "ROBOT", M1    'Speichert die Elementnummer 8 in M1
13 CHRSRCH C1$(1), "MELFA", M2    'Speichert die Elementnummer 2 in M2
```

Erläuterung

- Die festgelegt Zeichenkette wird in der Zeichenketten-Feldvariablen gesucht. Nur wenn die gesamte Zeichenkette dem Feldelement entspricht, erfolgt die Speicherung der Elementnummer. Teile der Feldelemente werden nicht gefunden. Die Ausführung der Anweisung ChrSrch C1$(1), "ROBO", M1 führt zu keinem Suchergebnis.
- Bleibt die Suche erfolglos, wird im Speicherort eine „0“ abgespeichert.
- Sie Suche der Zeichenkette erfolgt beginnend mit dem ersten Element sequentiell. Die Elementnummer des ersten übereinstimmenden Feldelements wird gespeichert. Bei der Suche über die Anweisung ChrSrch C1$(3), "MELFA", M2 im obigen Programmbeispiel wird M2 auf „2“ gesetzt, obwohl das Feldelement C1$(6) dieselbe Zeichenkette enthält.
- Es können nur eindimensionale Zeichenketten-Feldvariablen durchsucht werden. Beim Durchsuchen mehrdimensionaler Feldvariablen erfolgt eine Fehlermeldung.

6.3.6 Close (Close)

Funktion: Datei schließen

Schließt die festgelegte Datei (inklusive Kommunikationsleitungen).

Eingabeformat

Close □ [[#]<Dateinummer>[,[[#]<Dateinummer> ...]

<Dateinummer>	Legt die Dateinummer (1 bis 8) der zu schließenden Datei fest Es dürfen nur numerische Konstanten verwendet werden. Fehlt die Dateinummer, werden alle geöffneten Dateien geschlossen.

Programmbeispiel

```
1  Open "COM1:" AS#1          '„COM1:“ wird als Datei Nummer 1 geöffnet
2  PRINT #1,M1                'Überträgt den Inhalt von M1 in Datei Nummer 1
   :
10 INPUT #1,M2                'Liest die Daten von Datei Nummer 1 in M2 ein
11 CLOSE #1                   'Datei Nummer 1 schließen
   :
20 CLOSE                      'Alle geöffneten Dateien schließen
```

Erläuterung

- Die Close-Anweisung schließt alle Dateien (inklusive der Kommunikationsleitungen), die mit der Open-Anweisung geöffnet wurden. Die Daten aus dem Pufferspeicher werden entfernt.

Pufferspeicher	Datenverarbeitung beim Schließen von Dateien
Empfangspuffer der Kommunikationsleitung	Die Daten des Pufferspeichers werden gelöscht.
Sendepuffer der Kommunikationsleitung	Der Sendepuffer enthält keine Daten, da die Daten direkt nach Ausführung der Print-Anweisung verschickt wurden.
Pufferspeicher zum Laden von Dateien	Die Daten des Pufferspeichers werden gelöscht.
Pufferspeicher zum Speichern von Dateien	Die Daten des Pufferspeichers werden in die Datei geschrieben. Danach wird die Datei geschlossen.

Tab. 6-3: *Datenverarbeitung beim Schließen von Dateien*

- Durch die Ausführung der End-Anweisung wird eine Datei ebenfalls geschlossen.
- Fehlt die Dateinummer, werden alle geöffneten Dateien geschlossen.

Steht in Beziehung zu folgenden Befehlen:

Open, Print, Input

6.3.7 Clr (Clear)

Funktion: Löschfunktion

Zurücksetzen der allgemeinen Ausgänge, der lokalen und externen numerischen Variablen.

Eingabeformat

Clr ☐ <Ausführung>

<Ausführung> Die Ausführung kann als Konstante oder Variable vorgegeben werden.
0: Die allgemeinen Ausgangsbits, die lokalen und globalen Variablen werden zurückgesetzt.
1: Die allgemeinen Ausgänge werden auf ein voreingestelltes Bitmuster zurückgesetzt. Die Bitmuster werden über die Parameter ORST0 bis ORST224 vorgegeben (siehe auch Abschn. 9.15) (0: AUS, 1: EIN, *: HALTEN).
2: Alle lokalen numerischen Variablen und alle im Programm verwendeten numerischen Feldvariablen werden auf „0" gesetzt.
3: Alle externen numerischen Variablen (externe Systemvariablen und benutzerdefinierte externe Variablen) und externen numerischen Feldvariablen werden auf „0" gesetzt. Externe Positionsvariablen werden nicht zurückgesetzt.

Programmbeispiel

(1) Zurücksetzen der allgemeinen Ausgangssignale auf das vorgegebene Bitmuster

```
1 Clr 1                    'Zurücksetzen der allgemeinen Ausgangssignale
```

(2) Alle lokalen numerischen Variablen und alle im Programm verwendeten numerischen Feldvariablen werden auf „0" gesetzt.

```
1 Dim MA(10)               'Deklariert die Feldvariable MA als eine Variable
                           mit 10 Elementen
2 Def Inte IVAL            'Deklariert IVAL als Namen einer numerischen
                           Variablen
3 Clr 2                    'Setzt die Variablen MA(1) bis MA(10), IVAL und alle
                           lokalen numerischen Variablen des Programms auf „0"
```

(3) Alle externen numerischen Variablen und externen numerischen Feldvariablen werden auf „0" gesetzt.

```
1 CLR 3                    'Setzt alle externen numerischen Variablen und
                           Feldvariablen auf „0"
```

(4) Die Punkte (1) bis (3) werden gleichzeitig ausgeführt.

```
1 CLR 0                    'Die allgemeinen Ausgangsbits, die lokalen und
                           globalen Variablen werden zurückgesetzt.
```

Steht in Beziehung zu folgenden Parametern:

ORST0 bis ORST224

Steht in Beziehung zu folgenden Systemvariablen:

M_In/M_Inb/M_In8/M_Inw/M_In16, M_Out/M_Outb/M_Out8/M_Outw/M_Out16

6.3.8 Cmp Jnt (Compliance Joint)

Funktion: Achsenweichheit im Gelenk-Koordinatensystem aktivieren

Der Befehl legt fest, welche Achse im Gelenk-Koordinatensystem weich geschaltet werden soll. Er kann ausschließlich mit den Robotermodellen RH-3SDHR, RH-6/12/20SDH, RH-3SQHR und RH-6/12/20SQH verwendet werden.

Eingabeformat

Cmp □ JNT, <Achse>

<Achse>	Legt über ein Bitmuster die Achse fest, für die die Weichheit eingestellt werden soll (1: aktiv, 0: inaktiv) &B00000000: Achse 87654321

Programmbeispiel

```
1 Mov P1                  'Position P1 mittels Gelenk-Interpolation anfahren
2 CmpG 0.0,0.0,1.0,1.0,,,, 'Einstellung der Weichheit
3 Cmp Jnt, &B11           'Achsenweichheit für J1 und J2 aktivieren
4 Mov P2                  'Position P2 mittels Gelenk-Interpolation anfahren
5 HOpen 1                 'Öffnet Hand 1
6 Mov P1                  'Position P1 mittels Gelenk-Interpolation anfahren
7 Cmp Off                 'Achsenweichheit deaktivieren
```

Erläuterung

- Die Weichheit einer Roboterachse kann im Gelenk-Koordinatensystem festgelegt werden.
- Wird bei einem SCARA-Roboter beim senkrechten Einsetzen eines Bolzens in eine Bohrung die Weichheit für die Achsen J1 und J2 aktiviert, wird der Bolzen sanft in die Bohrung geführt (siehe Programmbeispiel oben).
- Der Wert der Weichheit entspricht einer Federkonstanten und wird über den Befehl CmpG eingestellt. Setzen Sie die Achsenweichheit bei einem SCARA-Roboter (z.B. RH-SDH) für die Achsen J1 und J2 auf „0.0“, um die Servowirkung bei der Steuerung des Roboters zu deaktivieren (servolose Steuerung, d.h. Federkonstante gleich 0). Eine servolose Steuerung der vertikalen Achsen ist nicht möglich, auch wenn der Wert auf „0.0“ eingestellt ist. Achten Sie darauf, dass die Achsen nicht außerhalb des zulässigen Bewegungsbereiches bewegt werden und dass die Positionsabweichung nicht zu groß wird.
- Die Einstellung der Weichheit bleibt auch bei einer Programmunterbrechung bis zur Ausführung des Befehls Cmp Off oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert.
- Ist die Weichheit aktiviert, kann der Roboter keine Position erreichen, die außerhalb des Verfahrwegbereiches der Gelenke liegt (nicht bei servolosem Betrieb).
- Ist die Abweichung der aktuellen Position von der Zielposition größer als 200 mm, unterbricht der Roboter die Verfahrbewegung und die Programmsteuerung springt zum nächsten Programmschritt (nicht bei servolosem Betrieb).

- Eine gleichzeitige Verwendung der Befehle Cmp Jnt, Pos und Tool ist nicht möglich. Soll z. B. nach Ausführung des Befehls CMP JNT einer der Befehle Cmp Pos oder Cmp Tool ausgeführt werden, erfolgt eine Fehlermeldung. Heben Sie zuerst die Einstellung der Weichheit über den Befehl Cmp Off auf, bevor Sie einen der Befehle ausführen.
- Schaltet sich bei aktivierter Weichheitseinstellung die Servospannung ein, kann sich die Position des Roboters ändern.
- Bei aktivierter Weichheitseinstellung ist ein JOG-Betrieb möglich. Die Einstellung der Weichheit kann nicht über die Teaching Box deaktiviert werden. Die Ausführung der Anweisung muss innerhalb eines Programms oder im Menü zur Programmeditierung erfolgen.
- Heben Sie zur Änderung der Achsenauswahl die Weichheitseinstellung über den Befehl Cmp Off auf und führen Sie anschließend den Befehl Cmp Jnt erneut aus.
- Die Einstellung der Weichheit ist nur für die Achsen des Roboterarms möglich. Bei Zusatzachsen (J7, J8 oder L1, L2) ist sie nicht wirksam, auch wenn sie eingestellt ist.
- Ist bei aktivierter Achsenweichheit gleichzeitig über den Fine-Befehl eine Bedingung zum Abschluss des Interpolationsbefehls festgelegt, kann es vorkommen, dass der Roboter den Bewegungsbefehl nicht so weit ausführt, dass die Ausgabe des Impulses bei Erreichen der Zielposition erfolgt. Der Roboter wartet auf die Beendigung des Befehls und die Programmausführung wird unterbrochen. Aktivieren Sie die Achsenweichheit nicht bei gleichzeitiger Verwendung der Feinpositionierung.

HINWEIS

Die Einstellung ist nur bei bestimmten Robotermodellen möglich. (Detaillierte Hinweise finden Sie im Technischen Handbuch des jeweiligen Roboters.)

ACHTUNG:

Führen Sie nach Aktivierung der Weichheitseinstellung über den Befehl Cmp Jnt den JOG-Betrieb im Gelenk-JOG-Modus aus. Die Wahl einer anderen JOG-Betriebsart kann aufgrund der unterschiedlichen Koordinatensysteme für die Weichheitseinstellung und den JOG-Betrieb zu Verfahrbewegungen in eine andere als die gewählte Richtung führen.

Beim Teachen von Positionen mit aktivierter Weichheitseinstellung muss zuerst die Servoversorgungsspannung ausgeschaltet werden. Ansonsten entspricht die geteachte Position nicht der aktuellen, sondern der ursprünglich geteachten Position.

Steht in Beziehung zu folgenden Systemvariablen:

M_CmpDst

Steht in Beziehung zu folgenden Befehlen:

Cmp Off, CmpG, Cmp Tool, Cmp Pos

6.3.9 Cmp Pos (Compliance Posture)

Funktion: Achsenweichheit im kartesischen Koordinatensystem aktivieren

Der Befehl legt fest, welche Achse im kartesischen Koordinatensystem weich geschaltet werden soll.

Eingabeformat

Cmp □ Pos, <Achse>

<Achse>	Legt über ein Bitmuster die Achse fest, für die die Weichheit eingestellt werden soll (1: aktiv, 0: inaktiv) &B00000000: L2, L1, C, B, A, Z, Y, X

Programmbeispiel

```
1 Mov P1                    'Position oberhalb der Einsetzposition mittels
                            Gelenk-Interpolation anfahren
2 CmpG 0.5,0.5,1.0,0.5,0.5,,, 'Einstellung der Weichheit
3 Cmp Pos, &B11011          'Achsenweichheit für die Achsen X, Y, A und B
                            aktivieren
4 Mvs P2                    'Einsetzposition mittels Linear-Interpolation
                            anfahren
5 M_Out(10) = 1             'Spannfutter für die Werkstückaufnahme schließen
6 Dly 1.0                   'Wartezeit von 1 s bis das Spannfutter geschlossen ist
7 HOpen 1                   'Öffnet Hand 1
8 Mvs, –100                 'Position anfahren, die 100 mm in Werkzeuglängsrichtung
                            von der aktuellen Position entfernt ist
9 Cmp Off                   'Achsenweichheit deaktivieren
```

Erläuterung

- Die Weichheit einer Roboterachse kann im Gelenk-Koordinatensystem festgelegt werden.
- Wird beim senkrechten Einsetzen eines Bolzens in eine Bohrung die Weichheit für die Achsen X, Y, A und B aktiviert, wird der Bolzen sanft in die Bohrung geführt.
- Der Wert der Weichheit wird über den Befehl CmpG eingestellt.
- Die Einstellung der Weichheit bleibt auch bei einer Programmunterbrechung bis zur Ausführung des Befehls Cmp Off oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert.
- Ist die Weichheit aktiviert, kann der Roboter keine Position erreichen, die außerhalb des Verfahrwegbereiches der Gelenke liegt.
- Die Abweichung der aktuellen Position von der Zielposition kann aus dem Parameter M_CmpDist ausgelesen werden. Der Parameter ermöglicht die Überprüfung des erfolgreichen Einsetzens eines Bolzens.
- Ist die Abweichung der aktuellen Position von der Zielposition größer als 200 mm, unterbricht der Roboter die Verfahrbewegung und die Programmsteuerung springt zum nächsten Programmschritt.
- Eine gleichzeitige Verwendung der Befehle Cmp Jnt, Pos und Tool ist nicht möglich. Soll z.B. nach Ausführung des Befehls Cmp Jnt einer der Befehle Cmp Pos oder Cmp Tool ausgeführt werden, erfolgt eine Fehlermeldung. Heben Sie zuerst die Einstellung der Weichheit über den Befehl Cmp Off auf, bevor Sie einen der Befehle ausführen.
- Schaltet sich bei aktivierter Weichheitseinstellung die Servospannung ein, kann sich die Position des Roboters ändern.
- Bei aktivierter Weichheitseinstellung ist ein JOG-Betrieb möglich. Die Einstellung der Weichheit kann nicht über die Teaching Box deaktiviert werden. Die Ausführung der Anweisung muss innerhalb eines Programms oder im Menü zur Programmeditierung erfolgen.
- Heben Sie zur Änderung der Achsenauswahl die Weichheitseinstellung über den Befehl Cmp Off auf und führen Sie anschließend den Befehl Cmp Pos erneut aus.
- Der Betrieb des Roboters in der Nähe eines singulären Punktes kann zu einer Fehlermeldung oder zu einem Fehler der Steuerung führen. Vermeiden Sie den Betrieb des Roboters in der Nähe singulärer Punkte.
  Tritt trotzdem einer dieser Fälle auf, deaktivieren Sie die Achsenweichheit bei ausgeschalteter Servoversorgungsspannung durch Ausführung des Befehls Cmp Off (oder schalten Sie die Versorgungsspannung aus und wieder ein), bewegen Sie den Roboter vom singulären Punkt weg und aktivieren Sie die Achsenweichheit erneut.
- Die Einstellung der Weichheit ist nur für die Achsen des Roboterarms möglich. Bei Zusatzachsen (J7, J8 oder L1, L2) ist sie nicht wirksam, auch wenn sie eingestellt ist.
- Ist bei aktivierter Achsenweichheit gleichzeitig über den Fine-Befehl eine Bedingung zum Abschluss des Interpolationsbefehls festgelegt, kann es vorkommen, dass der Roboter den Bewegungsbefehl nicht so weit ausführt, dass die Ausgabe des Impulses bei Erreichen der Zielposition erfolgt. Der Roboter wartet auf die Beendigung des Befehls und die Programmausführung wird unterbrochen. Aktivieren Sie die Achsenweichheit nicht bei gleichzeitiger Verwendung der Feinpositionierung.

Beispiel▽

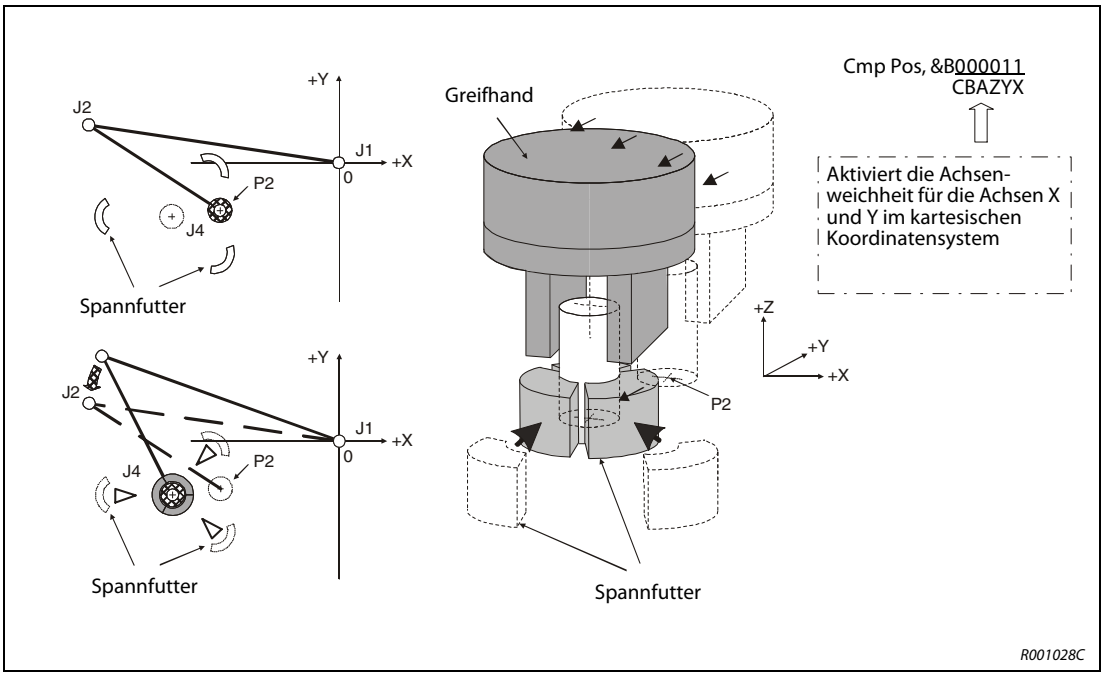


Abb. 6-3: *Achsenweichheit im kartesischen Koordinatensystem beim Einsetzen eines Bolzens in ein Spannfutter*

△

ACHTUNG:

Die Einstellung der Weichheit bleibt bis zur Ausführung des Befehls Cmp Off oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert. Besondere Vorsicht ist daher bei einem Programmwechsel und bei Ausführung des JOG-Betriebs geboten.

Führen Sie nach Aktivierung der Weichheitseinstellung über den Befehl Cmp Pos den JOG-Betrieb im XYZ-JOG-Modus aus. Die Wahl einer anderen JOG-Betriebsart kann aufgrund der unterschiedlichen Koordinatensysteme für die Weichheitseinstellung und den JOG-Betrieb zu Verfahrbewegungen in eine andere als die gewählte Richtung führen.

Beim Teachen von Positionen mit aktivierter Weichheitseinstellung muss zuerst die Servoversorgungsspannung ausgeschaltet werden. Ansonsten entspricht die geteachte Position nicht der aktuellen, sondern der ursprünglich geteachten Position.

Steht in Beziehung zu folgenden Systemvariablen:

M_CmpDst

Steht in Beziehung zu folgenden Befehlen:

Cmp Off, CmpG, Cmp Tool, Cmp Jnt

6.3.10 Cmp Tool (Compliance Tool)

Funktion: Achsenweichheit im Werkzeugkoordinatensystem aktivieren

Der Befehl legt fest, welche Achse im Werkzeugkoordinatensystem weich geschaltet werden soll.

Eingabeformat

Cmp ☐ Tool, <Achse>

<Achse>	Legt über ein Bitmuster die Achse fest, für die die Weichheit eingestellt werden soll (1: aktiv, 0: inaktiv) &B000000: C, B, A, Z, Y, X

Programmbeispiel

```
1 Mov P1                    'Position oberhalb der Einsetzposition mittels
                            Gelenk-Interpolation anfahren
2 CmpG 0.5,0.5,1.0,0.5,0.5,,, 'Einstellung der Weichheit
3 Cmp Tool, &B11011         'Achsenweichheit für die Achsen X, Y, A und B
                            aktivieren
4 Mvs P2                    'Einsetzposition mittels Linear-Interpolation
                            anfahren
5 M_Out(10) = 1             'Spannfutter für die Werkstückaufnahme schließen
6 Dly 1.0                   'Wartezeit von 1 s bis das Spannfutter geschlossen ist
7 HOpen 1                   'Öffnet Hand 1
8 Mvs, –100                 'Position anfahren, die 100 mm in Werkzeuglängsrichtung
                            von der aktuellen Position entfernt ist
9 Cmp Off                   'Achsenweichheit deaktivieren
```

Erläuterung

- Die Weichheit einer Roboterachse kann im Werkzeugkoordinatensystem festgelegt werden (siehe auch Abschn. 9.7).
- Wird beim senkrechten Einsetzen eines Bolzens in eine Bohrung die Weichheit für die Achsen X, Y, A und B aktiviert, wird der Bolzen sanft in die Bohrung geführt.
- Der Wert der Weichheit wird über den Befehl CmpG eingestellt.
- Die Einstellung der Weichheit bleibt auch bei einer Programmunterbrechung bis zur Ausführung des Befehls Cmp Off oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert.
- Ist die Weichheit aktiviert, kann der Roboter keine Position erreichen, die außerhalb des Verfahrwegbereiches der Gelenke liegt.
- Die Abweichung der aktuellen Position von der Zielposition kann aus dem Parameter M_CmpDist ausgelesen werden. Der Parameter ermöglicht die Überprüfung des erfolgreichen Einsetzens eines Bolzens.
- Ist die Abweichung der aktuellen Position von der Zielposition größer als 200 mm, unterbricht der Roboter die Verfahrbewegung und die Programmsteuerung springt zum nächsten Programmschritt.
- Eine gleichzeitige Verwendung der Befehle Cmp Jnt, Pos und Tool ist nicht möglich. Soll z. B. nach Ausführung des Befehls Cmp Jnt einer der Befehle Cmp Pos oder Cmp Tool ausgeführt werden, erfolgt eine Fehlermeldung. Heben Sie zuerst die Einstellung der Weichheit über den Befehl Cmp Off auf, bevor Sie einen der Befehle ausführen.
- Schaltet sich bei aktivierter Weichheitseinstellung die Servospannung ein, kann sich die Position des Roboters ändern.
- Bei aktivierter Weichheitseinstellung ist ein JOG-Betrieb möglich. Die Einstellung der Weichheit kann nicht über die Teaching Box deaktiviert werden. Die Ausführung der Anweisung muss innerhalb eines Programms oder im Menü zur Programmeditierung erfolgen.
- Heben Sie zur Änderung der Achsenauswahl die Weichheitseinstellung über den Befehl Cmp Off auf und führen Sie anschließend den Befehl Cmp Tool erneut aus.
- Bei fünfachsigen, vertikalen Knickarmrobotern (z. B. RV-3SDJB) kann die Weichheit im Werkzeugkoordinatensystem nur für die Achsen X und Z eingestellt werden.
- Der Betrieb des Roboters in der Nähe eines singulären Punktes kann zu einer Fehlermeldung oder zu einem Fehler der Steuerung führen. Vermeiden Sie den Betrieb des Roboters in der Nähe singulärer Punkte. Tritt trotzdem einer dieser Fälle auf, deaktivieren Sie die Achsenweichheit bei ausgeschalteter Servoversorgungsspannung durch Ausführung des Befehls Cmp Off (oder schalten Sie die Versorgungsspannung aus und wieder ein), bewegen Sie den Roboter vom singulären Punkt weg und aktivieren Sie die Achsenweichheit erneut.
- Die Einstellung der Weichheit ist nur für die Achsen des Roboterarms möglich. Bei Zusatzachsen ist sie nicht wirksam, auch wenn sie eingestellt ist.
- Ist bei aktivierter Achsenweichheit gleichzeitig über den Fine-Befehl eine Bedingung zum Abschluss des Interpolationsbefehls festgelegt, kann es vorkommen, dass der Roboter den Bewegungsbefehl nicht so weit ausführt, dass die Ausgabe des Impulses bei Erreichen der Zielposition erfolgt. Der Roboter wartet auf die Beendigung des Befehls und die Programmausführung wird unterbrochen. Aktivieren Sie die Achsenweichheit nicht bei gleichzeitiger Verwendung der Feinpositionierung.

Beispiel▽

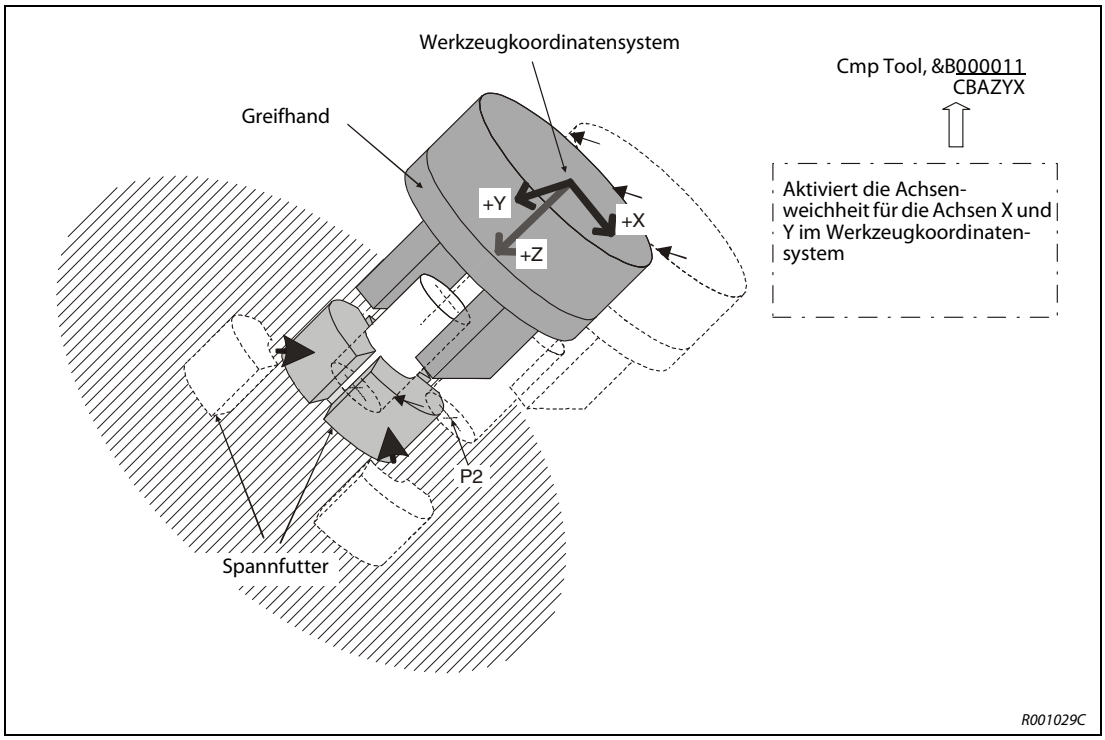


Abb. 6-4: *Achsenweichheit im Werkzeugkoordinatensystem beim Einsetzen eines Bolzens in ein Spannfutter*

△

ACHTUNG:

Die Einstellung der Weichheit bleibt bis zur Ausführung des Befehls Cmp Off oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert. Besondere Vorsicht ist daher bei einem Programmwechsel und bei Ausführung des JOG-Betriebs geboten.

Führen Sie nach Aktivierung der Weichheitseinstellung über den Befehl Cmp Tool den JOG-Betrieb im Werkzeug-JOG-Modus aus. Die Wahl einer anderen JOG-Betriebsart kann aufgrund der unterschiedlichen Koordinatensysteme für die Weichheitseinstellung und den JOG-Betrieb zu Verfahrbewegungen in eine andere als die gewählte Richtung führen.

Beim Teachen von Positionen mit aktivierter Weichheitseinstellung muss zuerst die Servoversorgungsspannung ausgeschaltet werden. Ansonsten entspricht die geteachte Position nicht der aktuellen, sondern der ursprünglich geteachten Position.

Steht in Beziehung zu folgenden Systemvariablen:

M_CmpDst

Steht in Beziehung zu folgenden Befehlen:

Cmp Off, CmpG, Cmp Pos, Cmp Jnt

6.3.11 Cmp Off (Compliance OFF)

Funktion: Achsenweichheit deaktivieren

Der Befehl deaktiviert die eingestellte Weichheit.

Eingabeformat

Cmp □ Off

Programmbeispiel

```
1 Mov P1                    'Position oberhalb der Einsetzposition mittels
                            Gelenk-Interpolation anfahren
2 CmpG 0.5,0.5,1.0,0.5,0.5, , ,  'Einstellung der Weichheit
3 Cmp Tool, &B11011         'Achsenweichheit für die Achsen X, Y, A und B
                            aktivieren
4 Mvs P2                    'Einsetzposition mittels Linear-Interpolation
                            anfahren
5 M_Out(10) = 1             'Spannfutter für die Werkstückaufnahme schließen
6 Dly 1.0                   'Wartezeit von 1 s bis das Spannfutter geschlossen ist
7 HOpen 1                   'Öffnet Hand 1
8 Mvs, –100                 'Position anfahren, die 100 mm in Werkzeuglängsrichtung
                            von der aktuellen Position entfernt ist
9 Cmp Off                   'Achsenweichheit deaktivieren
```

Erläuterung

- Die über die Befehle Cmp Tool, Cmp Pos oder Cmp Jnt eingestellte Weichheit wird deaktiviert.
- Deaktivieren Sie die Weichheitseinstellung im JOG-Betrieb, indem Sie die Anweisung in einem Programm oder im Menü zur Programmeditierung ausführen.

Steht in Beziehung zu folgenden Befehlen:

CmpG, Cmp Tool, Cmp Pos, Cmp Jnt

6.3.12 CmpG (Compliance Gain)

Funktion: Achsenweichheit einstellen

Der Befehl legt den Wert der Weichheit einer Achse fest.

Eingabeformat

Für die Befehle Cmp Pos und Cmp Tool

CmpG □ [<Einstellung X-Achse>], [<Einstellung Y-Achse>], [<Einstellung Z-Achse>], [<Einstellung A-Achse>], [<Einstellung B-Achse>], [<Einstellung C-Achse>],,

Für den Befehl Cmp Jnt

CmpG □ [<Einstellung J1-Achse>], [<Einstellung J2-Achse>], [<Einstellung J3-Achse>], [<Einstellung J4-Achse>], [<Einstellung J5-Achse>], [<Einstellung J6-Achse>],,

<Einstellung X- bis C-Achse>
<Einstellung J1- bis J6-Achse> Der Grad der Weichheit kann für jede Achse als Konstante eingestellt werden.
Die Einstellung „1" bedeutet Normalbetrieb, die Einstellung „0.2" entspricht der größten Weichheit.
Bei fehlender Angabe wird der aktuelle Wert verwendet.

Programmbeispiel

```
1 CmpG ,,0.5,,,,   'Einstellung der Weichheit erfolgt nur für die Z-Achse
                   Für die anderen Achsen bleiben die Stellen leer und
                   werden durch Kommas getrennt
```

Erläuterung

- Der Grad der Weichheit kann für jede Achse festgelegt werden.
- Die Aktivierung der Weichheit erfolgt über die Befehle Cmp Jnt, Cmp Pos oder Cmp Tool.
- Über den Befehl CmpG wird eine Kraft ähnlich einer Federkraft eingestellt, deren Größe von der Abweichung zwischen Zielposition und der aktuellen Position abhängig ist.
- Die Abweichung der aktuellen Position von der Zielposition kann aus dem Parameter M_CmpDist ausgelesen werden. Der Parameter ermöglicht die Überprüfung des erfolgreichen Einsetzens eines Bolzens.
- Bei kleinen Einstellwerten kann die Position bei Aktivierung der Weichheit durch den Befehl Cmp Jnt, Cmp Pos oder den Befehl Cmp Tool aufgrund des Eigengewichts des Roboterarms nach unten sinken. Stellen Sie die Achsenweichheit schrittweise ein.

- Bei aktivierter Achsenweichheit kann der Grad der Weichheit auch während des Betriebs verändert werden.
- Bei einer Weichheitseinstellung kleiner als der Minimalwert, wird der Minimalwert gesetzt.

Roboter	Cmp Pos, Cmp Tool	Cmp Jnt
RV-2SQ/6SQ/6SQL/12SQ, RV-2SD/6SD/6SDL/12SD	0.01, 0.01, 0.01, 0.01, 0.01, 0.01	Keine Einstellung möglich
RV-12SQL, RV-12SDL	0.01, 0.01, 0.05, 0.05, 0.01, 0.01	Keine Einstellung möglich
RV-3SQB/3SQJB, RV-2SDB, RV-3SDB/3SDBJ	0.10, 0.10, 0.10, 0.10, 0.10, 0.10	Keine Einstellung möglich
RH-3SDQR, RH-3SDHR, RH-6SQH/12SQH/20SQH, RH-6SDH/12SDH/20SDH	0.20, 0.20, 0.20, 0.20, 0.20, 0.20	0.00, 0.00, 0.20, 0.00, 1.00, 1.00

Tab. 6-4: *Minimalwert der Achsenweichheit*

- Die Einstellung der Weichheit ist nur für die Achsen des Roboterarms möglich. Bei Zusatzachsen (J7, J8 oder L1, L2) ist sie nicht wirksam, auch wenn sie eingestellt ist.

6.3.13 Cnt (Continuous)

Funktion: Roboterbewegung steuern

Der Befehl legt die Steuerung für eine kontinuierliche und gleichmäßige Bewegung fest und dient der Verkürzung der Zykluszeiten.

Eingabeformat

Cnt □ <Freigeben/Sperren> [,<Numerischer Wert 1>] [,<Numerischer Wert 2>]

<Freigeben/Sperren>	Legt den Anfang und das Ende einer kontinuierlichen und gleichmäßigen oder einer beschleunigten und abgebremsten Roboterbewegung fest 1 = freigeben, 0 = gesperrt
<Numerischer Wert 1>	Legt den Anfangspunktabstand der kontinuierlichen Bewegung in mm fest Der Standardwert ist der Startpunkt der Beschleunigung bzw. Abbremsung.
<Numerischer Wert 2>	Legt den Endpunktabstand der kontinuierlichen Bewegung in mm fest Der Standardwert ist der Startpunkt der Beschleunigung bzw. Abbremsung.

Programmbeispiel

```
1  Cnt 0                 'Sperren der Cnt-Einstellung
2  Mvs P1                'Position P1 mittels Linear-Interpolation und Beschleunigung/
                         Verzögerung anfahren
3  Cnt 1                 'Freigeben der Cnt-Einstellung
4  Mvs P2                'Position P2 mittels Linear-Interpolation und
                         kontinuierlicher gleichmäßiger Geschwindigkeit anfahren
5  Cnt 1, 100, 200       'Anfangspunktabstand der kontinuierlichen Bewegung auf
                         100 mm und Endpunktabstand der kontinuierlichen Bewegung
                         auf 200 mm festlegen
6  Mvs P3                'Position P3 mittels Linear-Interpolation und
                         kontinuierlicher gleichmäßiger Geschwindigkeit anfahren
7  Cnt 1, 300            'Anfangspunktabstand der kontinuierlichen Bewegung auf
                         300 mm und Endpunktabstand der kontinuierlichen
                         Bewegung auf 300 mm festlegen
8  Mov P4                'Position P4 mittels Gelenk-Interpolation und
                         kontinuierlicher gleichmäßiger Geschwindigkeit anfahren
9  Cnt 0                 'Sperren der Cnt-Einstellung
10 Mov P5                'Position P5 mittels Gelenk-Interpolation und
                         Beschleunigung/Verzögerung anfahren
```

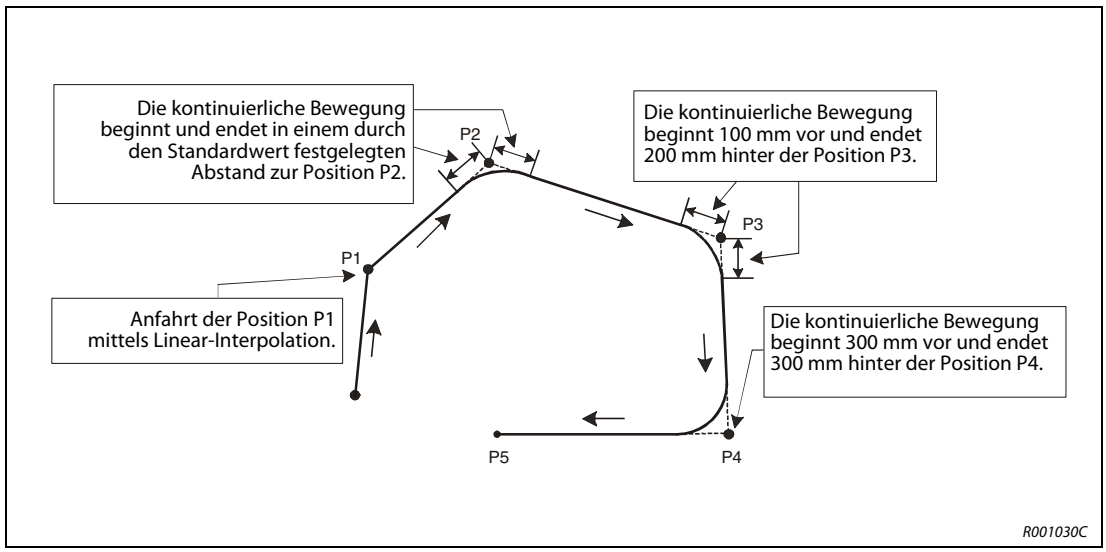

Die kontinuierliche Bewegung beginnt und endet in einem durch den Standardwert festgelegten Abstand zur Position P2.
P2
Die kontinuierliche Bewegung beginnt 100 mm vor und endet 200 mm hinter der Position P3.
P3
P1
Anfahrt der Position P1 mittels Linear-Interpolation.
Die kontinuierliche Bewegung beginnt 300 mm vor und endet 300 mm hinter der Position P4.
P5
P4
R001030C

Abb. 6-5: *Verfahrweg für die verschiedenen Cnt-Einstellungen*

ACHTUNG:

Der genaue Verfahrweg des Roboters kann sich in Abhängigkeit der Geschwindigkeit ändern. Besonders im Bereich der Eckpunkte kann der Verfahrweg variieren. Verfahren Sie den Roboter beim Start des Automatikbetriebs daher erst mit niedriger Geschwindigkeit. Steigern Sie die Geschwindigkeit allmählich und achten Sie darauf, dass keine Kollisionen mit umliegenden Einheiten auftreten.

Erläuterung

- Die Positionen zwischen den Befehlen Cnt 1 und Cnt 0 (Programmschritt 4 bis 8) werden mit kontinuierlicher gleichmäßiger Geschwindigkeit angefahren.
- Standardmäßig ist die Cnt-Einstellung gesperrt.
- Der Übergang zwischen den Verfahrwegsegmenten beginnt bei fehlender Angabe der numerischen Werte 1 und 2 am Startpunkt der Beschleunigung bzw. Abbremsung.
- Wie folgende Abbildung zeigt, wird die Verfahrbewegung bei deaktivierter Cnt-Einstellung vor der Zielposition bis zum Stopp abgebremst. Bei der Anfahrt der nächsten Position wird der Roboter erneut beschleunigt. Bei aktivierter Cnt-Einstellung erfolgt vor der Zielposition nur eine geringe Abbremsung des Roboters. Die Beschleunigung zur Anfahrt der nächsten Position setzt an diesem Punkt ein. Daher verläuft der Verfahrweg nicht durch jede Postion, sondern durch die Punkte, die durch den Anfangs- und Endpunktabstand festgelegt wurden.

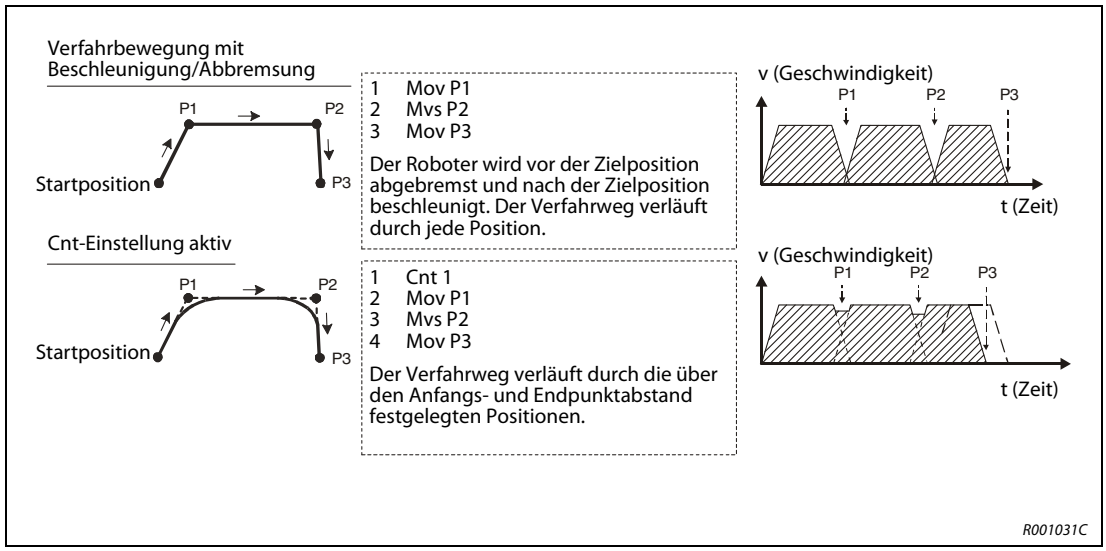


Abb. 6-6: *Einfluss der Cnt-Einstellung auf die Beschleunigung/Abbremsung*

- Die Anfangs- und Endpunktabstände definieren die Entfernungen der Punkte von der Zielposition, durch die der Verfahrweg verläuft. Der Übergang zwischen den Verfahrwegsegmenten beginnt bei fehlender Angabe der numerischen Werte 1 und 2 am Startpunkt der Beschleunigung bzw. Abbremsung. Die Zielposition wird dabei nicht durchlaufen, die Zykluszeit jedoch minimiert. Um die Abstände der Start- und Endposition der kontinuierlichen Bewegung zur Zielposition zu verkleinern, sind die numerischen Werte 1 und 2 zu verringern.

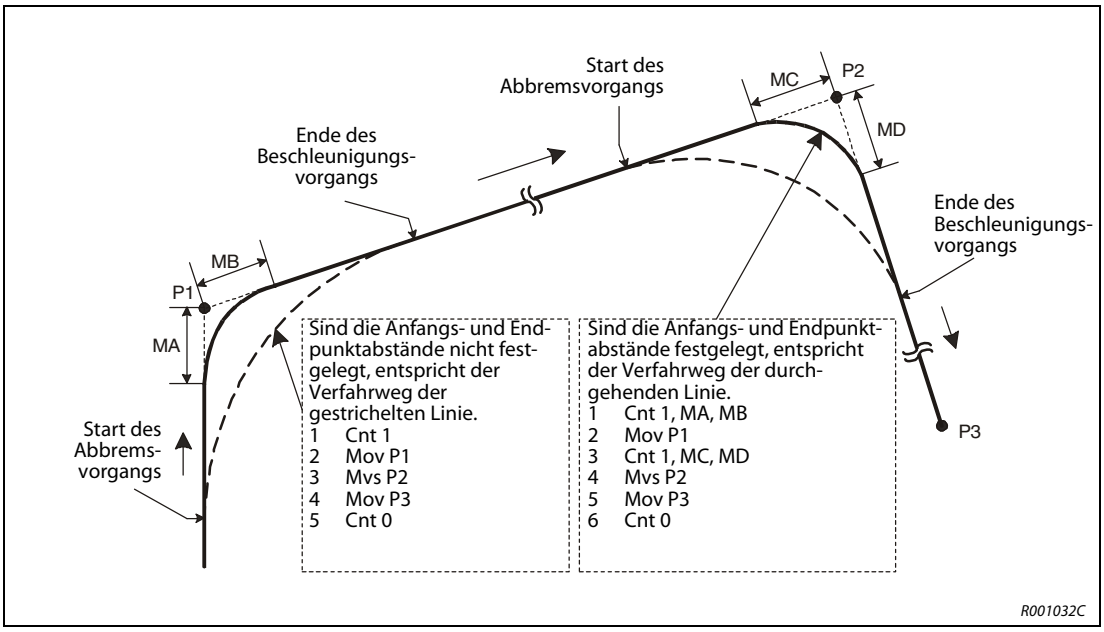


Abb. 6-7: *Festlegung von Anfangs- und Endpunktabstand*

- Fehlt die Angabe für den numerischen Wert 2, wird er auf den gleichen Wert wie der numerische Wert 1 gesetzt.
- Bei freigegebener Cnt-Einstellung ist die Fine-Einstellung gesperrt.
- Bei kleinen numerischen Werten kann das Durchlaufen des Verfahrweges länger dauern als bei deaktivierter Cnt-Einstellung.
- Auch wenn die Cnt-Einstellung aktiviert ist, wird der Interpolationsbefehl mit der Beschleunigung/Abbremsung ausgeführt, die der Einstellung „Durchfahren des singulären Punkts“ entspricht.

6.3.14 ColChk (Col Check)

Funktion: Kollisionsüberwachung aktivieren

Der Befehl aktiviert die Kollisionsüberwachung. Bei aktivierter Kollisionsüberwachung führt ein Zusammenstoß des Roboters mit umliegenden Einrichtungen zum sofortigen Stopp des Roboters. Die Funktion dient zur Begrenzung von Schäden, die bei einem Zusammenstoß entstehen können. Einen vollständigen Schutz vor Schäden und Verformungen an Komponenten des Roboters oder der umliegenden Einrichtungen bietet die Funktion jedoch nicht.

Eingabeformat

ColChk □ On[,NOErr]/Off

On	Aktiviert die Kollisionsüberwachung Bei einem Zusammenstoß erfolgt ein sofortiger Stopp des Roboters, die Fehlernummer 1010 wird ausgegeben und die Servos werden abgeschaltet.
Off	Deaktiviert die Kollisionsüberwachung
NOErr	Auch wenn die Kollisionsüberwachung anspricht, wird kein Fehler ausgegeben. Keine Angabe führt zur Ausgabe eines Fehlers.

Programmbeispiel 1

Bei einem Zusammenstoß erfolgt die Ausgabe einer Fehlermeldung

```
1 ColLvl 80,80,80,80,80,80,,   'Ansprechschwelle für die Kollisionsüberwachung festlegen
2 ColChk On                    'Kollisionsüberwachung aktivieren
3 Mov P1                       'Position P1 mittels Gelenk-Interpolation anfahren
4 Mov P2                       'Position P2 mittels Gelenk-Interpolation anfahren
5 Dly 0.2                      'Der Abschluss des Positioniervorgangs erfolgt über die
                               Timer-Einstellung. (Es kann auch die Anweisung Fine
                               verwendet werden.)
6 ColChk Off                   'Kollisionsüberwachung deaktivieren
7 Mov P3                       'Position P3 mittels Gelenk-Interpolation anfahren
```

Programmbeispiel 2

Bei einem Zusammenstoß erfolgt der Aufruf eines Interrupt-Prozesses

```
1  Def Act 1,M_ColSts(1) = 1 GoTo *HOME,S   'Definiert bei einem Zusammenstoß
                                            einen Unterprogrammsprung
                                            zur Marke HOME
2  Act 1 = 1                 'Interrupt 1 freigeben
3  ColChk On,NOErr           'Kollisionsüberwachung ohne Fehlerausgabe aktivieren
4  Mov P1
5  Mov P2                    'Erfolgt während der Ausführung der Schritte 4 bis 7 ein
                             Zusammenstoß, wird der Interrupt-Prozess ausgeführt
6  Mov P3
7  Mov P4
8  Act 1 = 0                 'Interrupt 1 sperren
   :
10 *HOME                     'Interrupt-Prozess bei einem Zusammenstoß
11 ColChk Off                'Kollisionsüberwachung deaktivieren
12 Servo On                  'Schaltet die Servospannung ein
13 PESC = P_ColDir(1)*(–2)   'Abstand der Ausweichposition festlegen
14 PDst = P_Fbc(1) + PESC    'Ausweichposition festlegen
15 Mvs PDst                  'Ausweichposition mittels Linear-Interpolation anfahren
16 Error 9100                'Benutzerdefinierten, leichten Fehler ausgeben
```

Erläuterung

- Die Auslösung der Kollisionsüberwachung erfolgt über eine Erfassung des Drehmomentes während der Ausführung eines Bewegungsbefehls. Als Zusammenstoß wird das Überschreiten eines bestimmten Differenzbetrages zwischen dem Drehmoment-Ist- und dem Drehmoment-Sollwert gewertet. Der Roboter wird sofort gestoppt.

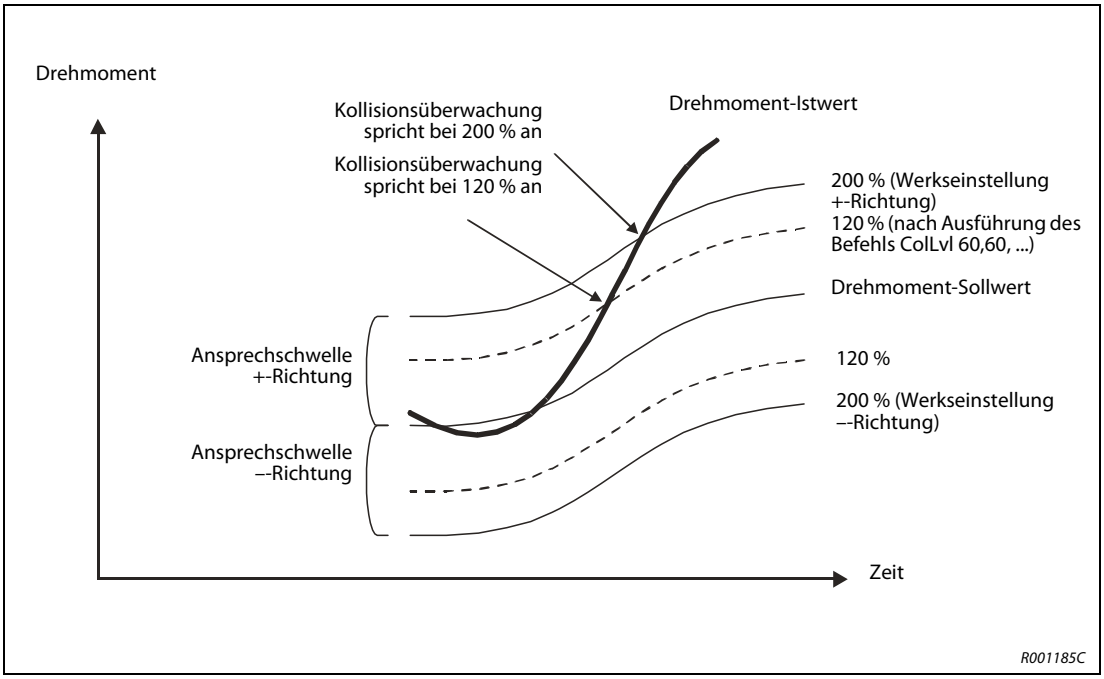


Abb. 6-8: *Ansprechschwelle der Kollisionsüberwachung*

- Direkt nach dem Einschalten der Spannungsversorgung ist die Kollisionsüberwachung deaktiviert. Geben Sie die Kollisionsüberwachung über den Parameter Col frei, bevor Sie sie verwenden. Über den Befehl kann die Funktion für den Programmbetrieb (einschließlich Schrittbetrieb und Programmsprung) aktiviert oder deaktiviert werden. Der Zustand der Funktion, wenn das Programm nicht ausgeführt wird, z. B. bei einer Pause oder im JOG-Betrieb, ist über das 3te Element des Parameters Col einzustellen.
- Die Ansprechschwelle der Kollisionsüberwachung kann über die Anweisung ColLvl eingestellt werden. Der Initialisierungswert ist im Parameter ColLvl festgelegt.
- Nach Aktivierung der Kollisionsüberwachung bleibt die Funktion solange aktiviert, bis die Anweisung ColChk Off oder End ausgeführt, das Programm zurückgesetzt oder die Spannungsversorgung ausgeschaltet wird.
- Auch wenn die Kollisionsüberwachung über die Anweisung wieder deaktiviert ist, bleibt die mit dem Befehl ColLvl eingestellte Ansprechschwelle erhalten.
- Bei einer kontinuierlichen Programmausführung wird die vorhergehende Einstellung der Kollisionsüberwachung auch dann übernommen, wenn die Spannungsversorgung aus und wieder eingeschaltet wird.
- Ist ein von der Roboterstatusvariablen M_ColSts abhängiger Interrupt (Interrupt-Bedingung: M_ColSts(*) = 1 mit Roboternummer = *) bei Definition des NOErr-Modus nicht freigegebenen (siehe Programmbeispiel 2), erfolgt die Ausgabe der Fehlermeldung 3950.
  Die Fehlermeldung 3960 wird ausgegeben, wenn der Interrupt bei aktiviertem NOErr-Modus gesperrt wird.

- Ein Zusammenstoß im NOErr-Modus führt zu einer Abschaltung der Servoversorgung und der Roboter stoppt. Es erfolgt keine Fehlermeldung und der Betrieb kann fortgesetzt werden. Der Zusammenstoß wird in einer LOG-Datei registriert, wenn nicht gleichzeitig andere Fehler auftreten.
- Wird eine der Anweisungen ColChk On oder ColChk On,NOErr auf einen Roboter angewendet, der nicht über die Funktion der Kollisionsüberwachung verfügt, erfolgt die Ausgabe des leichten Fehlers 3970. Bei der Anweisung ColChk Off erfolgt weder eine Fehlermeldung noch eine Verarbeitung der Anweisung.
- Eine Verwendung der Kollisionsüberwachung ist nicht möglich, wenn die Achsenweichheit über die Anweisung Cmp aktiviert oder eine Drehmomentgrenze über die Anweisung Torq definiert ist. Bei Aktivierung der Kollisionsüberwachung erfolgt in diesen Fällen die Ausgabe der Fehlermeldung 3940. Ist die Kollisionsüberwachung aktiviert, erfolgt bei einer Ausführung der Anweisung Cmp oder Torq die Ausgabe der Fehlermeldung 3930.
- Wird die Anweisung ColChk Off direkt nach einem Verfahrbefehl ausgeführt, ist die Kollisionsüberwachung kurz vor dem Erreichen der Zielposition eventuell nicht mehr wirksam. Fügen Sie daher zwischen dem Verfahrbefehl und der Anweisung ColChk Off eine Dly- oder Fine-Anweisung ein, um die Verfahrbewegung mit aktiver Kollisionsüberwachung abzuschließen (siehe Programmbeispiel 1).
- Eine falsche Einstellung der Parameter für die Hand- (HNDDATn) und Werkstückdaten (WRKDATn) kann zu einem ungewollten Ansprechen der Kollisionsüberwachung führen. Achten Sie daher auf eine korrekte Einstellung der Parameter.
- Eine Aktivierung der Kollisionsüberwachung kann in manchen Programmen zu einer Verlängerung der Zykluszeit führen. Aktivieren Sie die Funktion daher nicht für das gesamte Programm, sondern nur für Operationen, bei denen die Gefahr eines Zusammenstoßes besteht.
- Die Funktion kann nicht gemeinsam mit der Funktion zur Steuerung mehrerer Mechanismen verwendet werden.

Steht in Beziehung zu folgenden Befehlen:

ColLvl

Steht in Beziehung zu folgenden Systemvariablen:

M_ColSts, J_ColMxl, P_ColDir

Steht in Beziehung zu folgenden Parametern:

COL, COLLVL, COLLVLJG

6.3.15 ColLvl (Col Level)

Funktion: Ansprechschwelle der Kollisionsüberwachung

Der Befehl legt die Ansprechschwelle der Kollisionsüberwachung fest.

Eingabeformat

ColLvl □ [<J1-Achse>],[<J2-Achse>],[<J3-Achse>],[<J4-Achse>],[<J5-Achse>],[<J6-Achse>],,

<J1- bis J6-Achse>	Legt die Ansprechschwelle für jede Achse in einem Bereich zwischen 1 und 500 % fest. Erfolgt keine Angabe, ist der zuletzt eingestellte Wert wirksam. Für die Achsen J7 und J8 ist keine Einstellung möglich. Der Initialisierungswert ist im Parameter ColLvl festgelegt.

Programmbeispiel

```
1 ColLvl 80,80,80,80,80,80,,   'Ansprechschwelle für die Kollisionsüberwachung festlegen
2 ColChk On                    'Kollisionsüberwachung aktivieren
3 Mov P1                       'Position P1 mittels Gelenk-Interpolation anfahren
4 ColLvl ,50,50,,,,,           'Ansprechschwelle für die Kollisionsüberwachung für die Achsen
                               J2 und J3 ändern
5 Mov P2                       'Position P2 mittels Gelenk-Interpolation anfahren
6 Dly 0.2                      'Nach Erreichen der Position P2 wird die Kollisionsüberwachung
                               deaktiviert.
7 ColChk Off                   'Kollisionsüberwachung deaktivieren
8 Mov P3                       'Position P3 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Stellen Sie zum Programmbetrieb die Ansprechschwelle für die Kollisionsüberwachung auf einen für die jeweilige Achse zulässigen Wert ein.
- Dieser Befehl beeinflusst die Kollisionsüberwachung im Automatikbetrieb (einschließlich Schrittbetrieb und Programmsprung). Wenn das Programm nicht ausgeführt wird, z. B. bei einer Pause oder im JOG-Betrieb, ist der im Parameter ColLvlJG festgelegte Wert wirksam.
- Nach Einschalten der Spannungsversorgung sind die im Parameter ColLvl eingestellten Werte wirksam. Die Werkseinstellung des Parameters ist vom Robotermodell abhängig.
- Je größer der eingestellte Wert, desto höher die Ansprechschwelle und desto niedriger somit die Ansprechempfindlichkeit.
- Stellen Sie die Ansprechschwelle nicht zu niedrig ein, da dies zu einem ungewollten Ansprechen der Kollisionsüberwachung führen kann. In Abhängigkeit der Geschwindigkeit und der Roboterstellung ist eine ungewollte Auslösung auch bei Verwendung des Standardwertes möglich. Erhöhen Sie in diesem Falle die Ansprechschwelle.
- Eine falsche Einstellung der Parameter für die Hand- (HNDDATn) und Werkstückdaten (WRKDATn) kann zu einem ungewollten Ansprechen der Kollisionsüberwachung führen. Achten Sie daher auf eine korrekte Einstellung der Parameter.
- Bei einer kontinuierlichen Programmausführung wird die vorhergehende Einstellung der Kollisionsüberwachung auch dann übernommen, wenn die Spannungsversorgung aus und wieder eingeschaltet wird.
- Beim Zurücksetzen des Programmes oder bei Ausführung der Anweisung End wird die eingestellte Ansprechschwelle auf den im Parameter ColLvl festgelegten Wert gesetzt.
- Es erfolgt keine Fehlermeldung, wenn die Anweisung ColLvl auf einen Roboter angewendet wird, der nicht über die Funktion der Kollisionsüberwachung verfügt.
- Die Kollisionsüberwachung kann nicht auf die Achsen J7 und J8 angewendet werden.
- Die Einstellung ist vom jeweiligen Roboter abhängig. Auch wenn Sie Roboter des gleichen Typs verwenden, probieren Sie die passende Einstellung individuell mit jedem Roboter aus.

Steht in Beziehung zu folgenden Befehlen:

ColChk

Steht in Beziehung zu folgenden Systemvariablen:

M_ColSts, J_ColMxl, P_ColDir

Steht in Beziehung zu folgenden Parametern:

COL, COLLVL, HNDDATn, WRKDATn

6.3.16 Com Off (Communication OFF)

Funktion: Kommunikations-Interrupt sperren

Sperrt die Interrupts von Kommunikationskanälen

Eingabeformat

Com [(<Nummer des Kommunikationskanals>)] □ Off

<Nummer des Kommunikationskanals>	Legt die Nummer des Kommunikationskanals fest (z.B. 1, 2 oder 3)

Programmbeispiel

Ein Programmbeispiel finden Sie unter dem Befehl On Com GoSub in Abschn. 6.3.58.

Erläuterung

- Nach Ausführung des Com-Off-Befehls bleibt der Interrupt auch bei anstehenden Nachrichten gesperrt.
- Informationen über Kommunikationskanäle sind unter dem Befehl Open in Abschn. 6.3.61 zu finden.

6.3.17 Com On (Communication ON)

Funktion: Kommunikations-Interrupt freigeben

Gibt die Interrupts von Kommunikationskanälen frei

Eingabeformat

Com [(<Nummer des Kommunikationskanals>)] ☐ On

<Nummer des Kommunikationskanals>	Legt die Nummer des Kommunikationskanals fest (z.B. 1, 2 oder 3)

Programmbeispiel

Ein Programmbeispiel finden Sie unter dem Befehl On Com GoSub in Abschn. 6.3.58.

Erläuterung

- Informationen über Kommunikationskanäle sind unter dem Befehl Open in Abschn. 6.3.61 zu finden.

6.3.18 Com Stop (Communication STOP)

Funktion: Kommunikations-Interrupt stoppen

Stoppt die Interrupts von Kommunikationskanälen

Eingabeformat

Com [(<Nummer des Kommunikationskanals>)] □ Stop

<Nummer des Kommunikationskanals>	Legt die Nummer des Kommunikationskanals fest (z.B. 1, 2 oder 3)

Programmbeispiel

Ein Programmbeispiel finden Sie unter dem Befehl On Com GoSub in Abschn. 6.3.58.

Erläuterung

- Nach Ausführung des Com-Stop-Befehls wird der Interrupt auch bei anstehenden Nachrichten nicht generiert. Die anstehenden Daten und der Interrupt werden aufgezeichnet und beim nächsten Öffnen des Kanals abgearbeitet.
- Informationen über Kommunikationskanäle sind unter dem Befehl Open in Abschn. 6.3.61 zu finden.

6.3.19 Def Act (Define act)

Funktion: Interrupt-Prozess definieren

Dieser Befehl legt die Bedingungen eines Interrupts fest. Dazu gehört die gleichzeitige Überwachung von Signalen, die Interrupt-Verarbeitung während der Programmausführung und die Definition des Interrupt-Prozesses, der bei Auftreten eines Interrupts ausgeführt werden soll.

Eingabeformat

Def ☐ Act ☐ <Priorität>,<Ausdruck> ☐ <Prozess>[,<Typ>]

<Priorität>	Gibt die Priorität des Interrupts an 1 ≤ Priorität ≤ 8
<Ausdruck>	Folgende Formate können für den Interrupt-Status verwendet werden (siehe auch Syntaxdiagramm): <Num. Datentyp> <Vergleichsoperator> <Num. Datentyp> oder <Num. Datentyp> <Logischer Operator> <Num. Datentyp>. Die Angabe <Numerischer Datentyp> bezieht sich auf: <Numerische Konstanten>|<Numerische Variablen>|<Numerische Feldvariablen>|<Komponentendaten>.
<Prozess>	Legt eine GoTo- oder GoSub-Anweisung fest, die bei einem Interrupt ausgeführt wird
<Typ>	Bei fehlender Angabe: Stoppmethode 1 Die Stoppposition des Roboters ergibt sich bei einer Geschwindigkeitsübersteuerung von 100 %. Wird der Wert verkleinert, ist die Zeit bis zum Stopp länger, die Stoppposition jedoch bleibt dieselbe. S: Stoppmethode 2 Unabhängig von der Einstellung der Geschwindigkeitsübersteuerung stoppt der Roboter in der kürzestmöglichen Zeit. L: Stopp nach Abarbeitung des Programmschritts Die Angabe „L“ legt fest, dass der Interrupt-Prozess nach Erreichen der Zielposition (Abarbeitung des Programmschritts) ausgeführt wird.

Programmbeispiel

```
1  Def Act 1,M_In(17) = 1 GoSub *SUB1      'Definiert einen Unterprogrammsprung
                                           zur Marke SUB1, wenn der Status
                                           des allgemeinen Eingangssignals
                                           Nummer 17 = EIN ist
2  Def Act 2,MFG1 AND MFG2 GoTo *L200      'Definiert einen Programmsprung
                                           zur Marke L200, wenn das Resultat
                                           der UND-Verknüpfung von MFG1 und
                                           MFG2 „wahr“ ist
3  Def Act 3,M_Timer(1) > 10.5 GoSub *LBL  'Definiert nach Ablauf von 10,5 s
                                           einen Unterprogrammsprung zur
                                           Marke LBL
   :
9  *SUB1                                   'Marke SUB1 festgelegt
10 M_Timer(1) = 0                          'Zähler zurücksetzen
11 Act 3 = 1                               'Interrupt 3 freigeben
12 Return 0                                'Sprung zu dem Programmschritt,
                                           von dem der Interrupt aufgerufen
                                           wurde
19 *L200                                   'Marke L200 festgelegt
20 Mov P_SAFE                              'Rückzugspunkt mittels Gelenk-
                                           Interpolation anfahren
21 End                                     'Unterprogrammende
   :
30 *LBL                                    'Marke LBL festgelegt
31 M_Timer(1) = 0                          'Zähler zurücksetzen
32 Act 3 = 0                               'Interrupt 3 sperren
33 Return 0                                'Sprung zu dem Programmschritt, von
                                           dem der Interrupt aufgerufen wurde
```

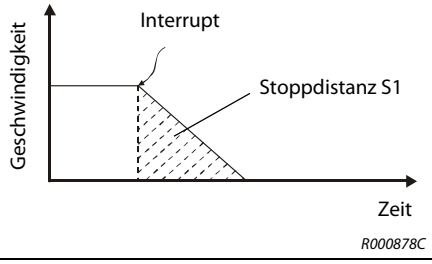
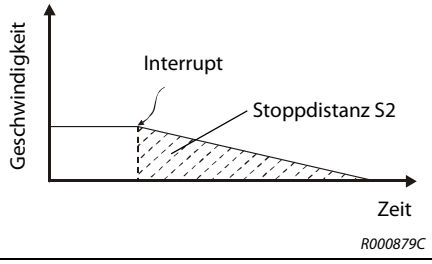
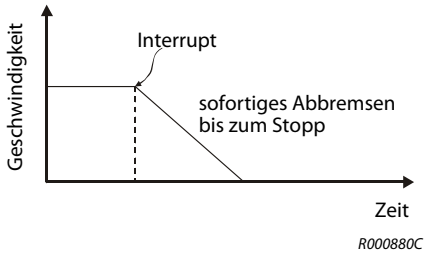
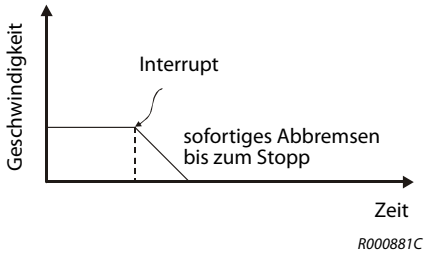
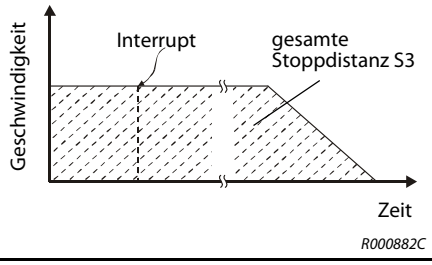
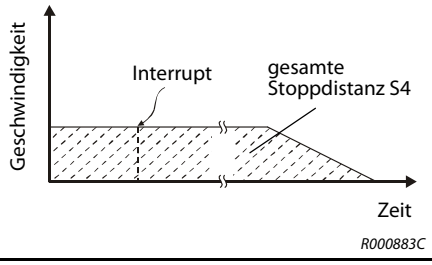
Erläuterung

- Die Prioritäten der Interrupts sind in aufsteigender Reihenfolge von 1 bis 8 festgelegt.
- Über die Priorität können bis zu 8 Interrupts unterschieden werden.
- Als Ausdruck dürfen einfache logische Operationen oder Vergleichsoperationen (ein Operand) verwendet werden. Klammerausdrücke sind nicht zulässig.
- Haben zwei Interrupts dieselbe Priorität, ist der später definierte Interrupt vorrangig.
- Der Def-Act-Befehl definiert nur den Interrupt. Mit dem Act-Befehl wird der Status des Interrupts festgelegt.
- Der Kommunikations-Interrupt (COM) hat eine höhere Priorität als Interrupts, die mit dem Def-Act-Befehl definiert wurden.
- Def-Act-Definitionen sind nur in dem Programm wirksam, in dem sie definiert wurden. In einem Unterprogramm müssen sie gegebenenfalls neu definiert werden.
- Wird ein Interrupt durch eine GoTo-Anweisung in einem Def-Act-Befehl generiert, bleibt der Interrupt während der Abarbeitung des verbleibenden Programmteils erhalten und es werden nur Interrupts höherer Priorität akzeptiert. Der Interrupt kann durch die Ausführung der End-Anweisung deaktiviert werden.
- Ausdrücke mit Kombinationen aus logischen Operationen und Vergleichsoperationen wie z. B. (M1 AND &H001) = 1 sind nicht zulässig.

ACHTUNG:

Wählen Sie die Stoppmethode passend zum Anwendungszweck. Soll z. B. die Ausführung eines Bewegungsbefehls in kürzester Zeit mit kürzestem Weg gestoppt werden, wählen Sie die Einstellung „S“.

Folgende Diagramme zeigen die unterschiedlichen Stoppmethoden bei Unterbrechung einer Verfahrbewegung durch einen Interrupt-Prozess.

Stoppmethode	Übersteuerung 100 %	Übersteuerung 50 %
Stoppmethode 1 (bei fehlendem Argument) S1 = S2		
Stoppmethode 2 (S)		
Stopp nach Abarbeitung des Programmschritts (L) S3 = S4		

Tab. 6-5: *Stoppmethoden*

Steht in Beziehung zu folgenden Befehlen:

Act

6.3.20 Def Arch (Define Arch)

Funktion: Bogen definieren

Definiert einen Bogen für die Verfahrbewegung mittels Bogen-Interpolation über den Befehl Mva.

Eingabeformat

```
Def Arch □ <Bogennummer>,
           [<Schrittweite bei Aufwärtsbewegung>],
           [<Schrittweite bei Abwärtsbewegung>],
           [<Hilfsschrittweite bei Aufwärtsbewegung>],
           [<Hilfsschrittweite bei Abwärtsbewegung>],
           [<Interpolationstyp>],
           [<Interpolationstyp 1>,<Interpolationstyp 2>]
```

<Bogennummer>	Gibt die Nummer des Bogens als Konstante oder Variable an 1 ≤ Bogennummer ≤ 4
<Schrittweite bei Aufwärtsbewegung> <Schrittweite bei Abwärtsbewegung> <Hilfsschrittweite bei Aufwärtsbewegung> <Hilfsschrittweite bei Abwärtsbewegung>	Legt die Schrittweite als Konstante oder Variable fest (siehe folgende Abbildung)

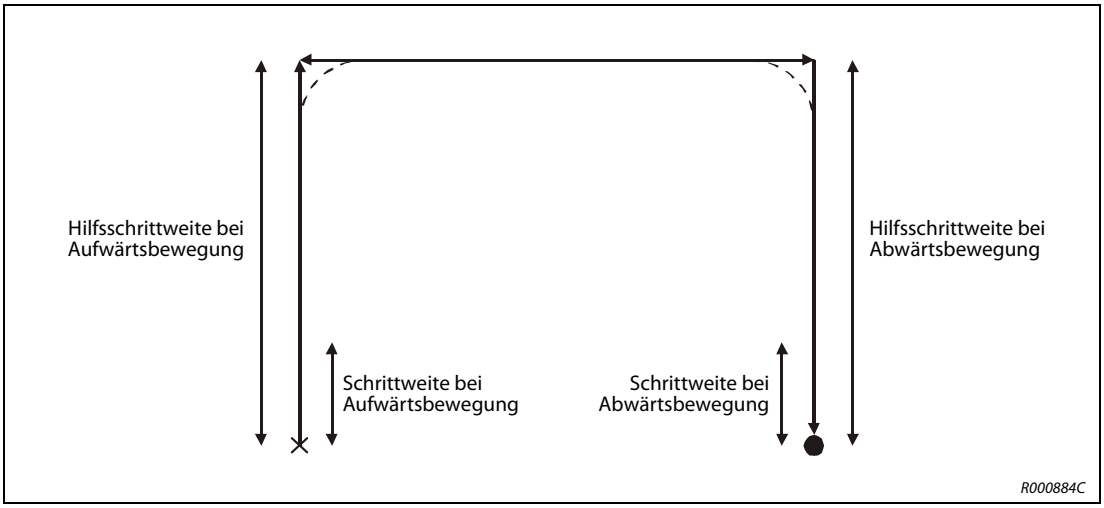


Abb. 6-9: *Definition des Bogens*

<Interpolationstyp>	Legt die Interpolationsart für die Aufwärts- und Abwärtsbewegungen fest Linear/Gelenk = 1/0
<Interpolationstyp 1>	Indirekt/direkt = 1/0
<Interpolationstyp 2>	3-Achsen-XYZ/äquivalente Drehung = 1/0

Für fehlende Argumente neben der Bogennummer werden die Standardwerte gesetzt. Die Standardwerte sind über folgende Parameter festgelegt. Prüfen Sie die Werte der Parameter vor ihrer Verwendung. Die Parameterwerte sind veränderbar.

Parameter	Bogen-nummer	Schrittweite bei Aufwärts-bewegung [mm]	Schrittweite bei Abwärts-bewegung [mm]	Hilfsschrittweite bei Aufwärts-bewegung [mm]	Hilfsschrittweite bei Abwärts-bewegung [mm]
ARCH1S	1	0.0	0.0	30.0	30.0
ARCH2S	2	10.0	10.0	30.0	30.0
ARCH3S	3	20.0	20.0	30.0	30.0
ARCH4S	4	30.0	30.0	30.0	30.0

Tab. 6-6: *Standardwerte der Bogenparameter (Schrittweiten)*

Für die Interpolationstypen gelten folgende Standardwerte:

Parameter	Bogen-nummer	Vertikaler Knickarmroboter (RV-6SDL usw.)			SCARA-Roboter (RH-12SDH usw.)		
		Inter-polationstyp	Interpola-tionstyp 1	Interpola-tionstyp 2	Inter-polationstyp	Interpola-tionstyp 1	Interpola-tionstyp 2
ARCH1T	1	1	0	0	0	0	0
ARCH2T	2	1	0	0	0	0	0
ARCH3T	3	1	0	0	0	0	0
ARCH4T	4	1	0	0	0	0	0

Tab. 6-7: *Standardwerte der Bogenparameter (Interpolationstypen)*

Programmbeispiel

```
1 Def Arch 1,5,5,20,20        'Definiert den Bogen
2 Mva P1,1                    'Position P1 mittels Bogen-Interpolation über den
                               in Programmschritt 1 definierten Bogen anfahren
3 Dly 0.3                     'Wartezeit von 0,3 Sekunden
4 Mva P2,2                    'Position P2 mittels Bogen-Interpolation über den mit
                               den Standardeinstellungen definierten Bogen anfahren
5 Dly 0.3                     'Wartezeit von 0,3 Sekunden
```

Erläuterung

- Wird der Befehl Mva ohne vorherige Definition des Bogens über den Def-Arch-Befehl ausgeführt, erfolgt die Verfahrbewegung entsprechend dem über die Parameter festgelegten Bogen.
- Der Befehl ermöglicht die Ausführung von Bewegungsbefehlen mit unterschiedlichen Schrittweiten in einem Programm.

ACHTUNG:

Der genaue Verfahrweg des Roboters kann sich in Abhängigkeit der Geschwindigkeit ändern. Besonders im Bereich der Eckpunkte kann der Verfahrweg variieren. Verfahren Sie den Roboter beim Start des Automatikbetriebs daher erst mit niedriger Geschwindigkeit. Steigern Sie die Geschwindigkeit allmählich und achten Sie darauf, dass keine Kollisionen mit umliegenden Einheiten auftreten.

Steht in Beziehung zu folgenden Befehlen:

Mva, Accel, Ovrd, Mvs (Erläuterung der Funktion „TYPE“)

6.3.21 Def Char (Define Character)

Funktion: Zeichenkettenvariable definieren

Dieser Befehl definiert eine Zeichenkettenvariable. Er wird dann zur Variablendeklaration verwendet, wenn der Variablenname mit einem anderen Buchstaben als mit „C“ beginnen soll. Variablennamen, die mit „C“ beginnen, müssen nicht über den Befehl Def Char deklariert werden.

Eingabeformat

```
Def □ Char □ <Name der Zeichenkettenvariablen>
             [,<Name der Zeichenkettenvariablen>] ...
```

<Name der Zeichenkettenvariablen> Legt den Namen der Zeichenkettenvariablen fest

Programmbeispiel

```
1 Def Char MESSAGE          'Deklariert „MESSAGE“ als Namen einer Zeichenkettenvariablen
2 MESSAGE = "WORKSET"       'Schreibt die Zeichenkette „WORKSET“ in die
                            Zeichenkettenvariable MESSAGE
3 CMSG = "ABC"              'Schreibt die Zeichenkette „ABC“ in die
                            Zeichenkettenvariable CMSG
                            Zeichenkettenvariablen, deren Variablenname mit „C“ beginnt,
                            müssen nicht über den Befehl Def Char deklariert werden
```

Erläuterung

- Der Variablenname darf aus maximal 16 Zeichen bestehen. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 240 Zeichen (inklusive des Befehls) in einem Programmschritt verwendet werden.
- Durch einen Unterstrich „_“ hinter dem „C“ wird eine im Basisprogramm definierte Variable zur globalen Variablen und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.22 Def Fn (Define function)

Funktion: Funktion definieren

Definiert eine Funktion und legt den Namen fest

Eingabeformat

```
Def □ Fn  <Identifizierungszeichen> <Name> [(<Formalparameter>[,<Formalparameter>] ...)]
          = <Funktionsausdruck>
```

<Identifizierungszeichen>	Es werden vier Zeichen zur Identifizierung unterschieden: M: Typ Numerisch C: Typ Zeichenkette P: Typ Position J: Typ Gelenk
<Name>	Benutzerdefinierte Zeichenkette (max. 5 Zeichen)
<Formalparameter>	Legt die Variablen der Funktion fest Es können maximal 16 Variablen verwendet werden.
<Funktionsausdruck>	Legt die Rechenoperation fest.

Programmbeispiel

```
1 Def FNMAVE(MA,MB) = (MA+MB)/2   'Legt fest, dass durch FNMAVE der Durchschnitt
                                  von zwei numerischen Variablen gebildet wird
2 MDATA1 = 20                     'Weist MDATA1 den Wert 20 zu
3 MDATA2 = 30                     'Weist MDATA2 den Wert 30 zu
4 MAVE = FNMAVE(MDATA1,MDATA2)    'Der Durchschnitt von 20 und 30 (= 25) wird
                                  der numerischen Variablen MAVE zugewiesen.
5 Def FNPADD(PA,PB) = (PA+PB)     'Legt fest, dass durch FNPADD die Summe
                                  von zwei Positionsvariablen gebildet wird
6 P10 = FNPADD(P1,P2)             'Weist P10 die Summe von P1 und P2 zu
```

Erläuterung

- Durch FN und <Name> wird der Name der Funktion festgelegt. Der Funktionsname kann bis zu 8 Zeichen lang sein.
  Beispiel:
  Numerischer Typ ... FNMAX Identifizierungszeichen: M
  Zeichenkettentyp ... FNCAME$ Identifizierungszeichen: C (Wird durch „$“ abgeschlossen)
- Eine mit Def Fn definierte Funktion heißt benutzerdefinierte Funktion.
- Es können Funktionen bis zu maximal einer Programmschrittlänge beschrieben werden.
- Im <Funktionsausdruck> können fest definierte und schon vorher vom Benutzer definierte Funktionen verwendet werden. In diesem Fall können 16 Ebenen von benutzerdefinierten Funktionen verwendet werden.
- Wenn die Variablen im Funktionsausdruck nicht in den Formalparametern aufgeführt wurden, erfolgt für die Variablen eine Verarbeitung der augenblicklichen Werte. Es tritt eine Fehlermeldung auf, wenn die Anzahl oder der Typ der verwendeten Variablen (numerische oder Zeichenkette) von den deklarierten abweichen.
- Eine benutzerdefinierte Funktion steht nur in dem Programm zur Verfügung, in dem sie definiert worden ist. Sie kann von einem anderen Programm nicht durch einen CallP-Befehl aufgerufen werden.

6.3.23 Def Inte/Long/Float/Double (Define Integer/Long/Float/Double)

Funktion: Numerische Variable deklarieren

Dieser Befehl definiert eine numerische Variable. Dabei steht Inte für den Typ Integer, Long für den Typ Integer mit Doppelwort-Genauigkeit, Float für den Typ Real mit einfacher Genauigkeit und Double für den Typ Real mit doppelter Genauigkeit.

Eingabeformat

Def ☐ Inte ☐ <Name einer numerischen Variablen> [, <Name einer numerischen Variablen>] ...
Def ☐ Long ☐ <Name einer numerischen Variablen> [, <Name einer numerischen Variablen>] ...
Def ☐ Float ☐ <Name einer numerischen Variablen> [, <Name einer numerischen Variablen>] ...
Def ☐ Double ☐ <Name einer numerischen Variablen> [, <Name einer numerischen Variablen>] ...

<Name einer numerischen Variablen>	Legt den Variablennamen fest

Programmbeispiel

Deklaration einer Variablen vom Typ Integer

```
1  Def Inte WORK1, WORK2          'Deklariert WORK1 und WORK 2 als Namen zweier
                                  numerischer Variablen
2  WORK1 = 100                    'Setzt den Wert der numerischen Variablen WORK1
                                  auf „100“
3  WORK2 = 10.562                 'Setzt den Wert der numerischen Variablen WORK2
                                  auf „11“
4  WORK2 = 10.12                  'Setzt den Wert der numerischen Variablen WORK2
                                  auf „10“
```

Deklaration einer Variablen vom Typ Integer mit Doppelwort-Genauigkeit

```
1  Def Long WORK3                 'Deklariert WORK3 als Namen einer
                                  numerischer Variablen
2  WORK3 = 12345                  'Setzt den Wert der numerischen Variablen WORK3
                                  auf „12345“
```

Deklaration einer Variablen vom Typ Real mit einfacher Genauigkeit

```
1  Def Float WORK4                'Deklariert WORK4 als Namen einer numerischen
                                  Variablen
2  WORK4 = 123.468                'Setzt den Wert der numerischen Variablen WORK4
                                  auf „123,468000“
```

Deklaration einer Variablen vom Typ Real mit doppelter Genauigkeit

```
1  Def Double WORK5               'Deklariert WORK5 als Namen einer numerischen
                                  Variablen
2  WORK5 = 100/3                  'Setzt den Wert der numerischen Variablen WORK5
                                  auf „33,333332061767599“
```

Erläuterung

- Der Variablenname kann bis zu 16 Zeichen lang sein. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 240 Zeichen (inklusive Befehl) in einem Programmschritt verwendet werden.
- Eine mit dem Befehl Inte deklarierte Variable ist vom Typ Integer (–32768 bis +32767).
- Eine mit dem Befehl Long deklarierte Variable ist vom Typ Integer (–2147483648 bis +2147483647).
- Eine mit dem Befehl Float deklarierte Variable ist eine numerische Variable mit einfacher Genauigkeit (±1,70141E+38).
- Eine mit dem Befehl Double deklarierte Variable ist eine numerische Variable mit doppelter Genauigkeit (±1,701411834604692E+308).

6.3.24 Def IO (Define IO)

Funktion: Ein-/Ausgangsvariable definieren

Dieser Befehl definiert eine Ein-/Ausgangsvariable. Sie dient zur Festlegung von Bitbreiten. Die Variablen M_In und M_Out werden für Einzelbitsignale, die Variablen M_Inb und M_Outb für 8-Bit breite Bytes, die Variablen M_Inw und M_Outw für 16-Bit breite Wörter und die Variablen M_In32 und M_Out32 für 32-Bit breite Wörter verwendet.

Eingabeformat

Def ☐ IO ☐ <E/A-Variablenname> = <Variablentyp>, <E/A-Bitnummer>][,<Maskeninformation>]

<E/A-Variablenname>	Legt den Variablennamen fest
<Variablentyp>	Legt fest, ob die Variable vom Typ BIT (1 Bit), BYTE (8 Bit), WORD (16 Bit) oder INTEGER ist
<E/A-Bitnummer>	Legt die Nummer des Eingangs-/Ausgangsbits fest
<Maskeninformation>	Legt fest, ob ein bestimmtes Signal zugelassen wird

Programmbeispiel

Zuweisung des Ein-/Ausgangsbits Nr. 6 vom Typ BIT an die Ein-/Ausgangsvariable mit dem Namen PORT1

```
1  Def IO PORT1 = BIT,6          'Weist der Ein-/Ausgangsvariablen mit dem
   :                             Namen PORT1 das Ein-/Ausgangsbit Nr. 6 vom
   :                             Typ BIT zu
10 PORT1 = 1                     'Schaltet Ausgangssignal 6 ein
   :
20 PORT1 = 2                     'Schaltet Ausgangssignal 6 aus (da die niedrigste
                                 Stelle des numerischen Werts „2“ gleich „0“ ist
21 M1 = PORT1                    'Weist der Variablen M1 den Wert des
                                 Eingangssignals 6 zu
```

Zuweisung des Ein-/Ausgangsbits Nr. 5 vom Typ BYTE an die Ein-/Ausgangsvariable mit dem Namen PORT2 und Festlegung der Maskeninformation auf hexadezimal „0F“

```
1  Def IO PORT2 = BYTE,6,&H0F    'Weist der Ein-/Ausgangsvariablen mit dem
                                 Namen PORT2 das Ein-/Ausgangsbit Nr. 6 vom
   :                             Typ BYTE zu und legt die Maske auf hexadezimal
                                 „0F“ fest
10 PORT2 = &HFF                  'Schaltet die Ausgangssignale 5 bis 8 ein
   :
20 M2 = PORT2                    'Weist der Variablen M2 den Wert der
                                 Eingangssignale 5 bis 8 zu
```

Zuweisung des Ein-/Ausgangsbits Nr. 8 vom Typ WORD an die Ein-/Ausgangsvariable mit dem Namen PORT3 und Festlegung der Maskeninformation auf hexadezimal „0FFF“

```
1  Def IO PORT3 = WORD,8,&H0FFF      'Weist der Ein-/Ausgangsvariablen mit dem
                                     Namen PORT3 das Ein-/Ausgangsbit Nr. 8
   :                                 vom Typ WORD zu und legt die Maske
                                     auf hexadezimal „0FFF“ fest
10 PORT3 = 9                         'Schaltet die Ausgangssignale 8 und 11 ein
   :
20 M3 = PORT3                        'Weist der Variablen M3 den Wert der
                                     Eingangssignale 8 bis 19 zu
```

Erläuterung

- Eingangssignale werden beim Zugriff auf die Variable eingelesen.
- Ausgangssignale werden geschrieben, wenn der Variablen ein Wert zugewiesen wird.
- Über die mit dem Befehl Def IO definierten Variablen kann kein Zugriff auf die Ausgangssignale erfolgen. Verwenden Sie zu diesem Zweck die Variable M_Out.
- Der Variablenname kann bis zu 16 Zeichen lang sein. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Ist eine Maske angegeben, wird nur ein bestimmtes Signal zugelassen.

Beispiel▽

In Programmschritt 20 wird bei einem 8 Bit breiten Byte ein Maskierungsprozess mit dem hexadezimalen Wert 0F ausgeführt.
Für PORT 2 ergibt sich danach als

- Eingangssignal (M1 = PORT2), dass die Bits Nummer 5 bis 8 als Eingänge zugelassen und die Bits Nummer 9 bis 12 auf „0“ gesetzt werden

```
   gesperrt      zugelassen
0   0   0   0   1   1   1   1
12                          5 (E/A-Bitnummer)
```

- Ausgangssignal (PORT2 = M1), dass die Bits Nummer 5 bis 8 die aktuellen Daten ausgeben und die Bits Nummer 9 bis 12 die Signalzustände beibehalten

```
Daten beibehalten   aktuelle Daten ausgeben
      *   *   *   *  1   1   1   1
     12                        5 (E/A-Bitnummer)
```

△

Beispiel▽

Mit Hilfe der Def-IO-Funktion ist es möglich, sowohl einen Eingang als auch einen Ausgang zu definieren. Generell lässt sich die Funktion wie folgt beschreiben: Wird die Variable auf der linken Seite eines mathematischen Ausdrucks benutzt, wird sie als Ausgang festgelegt. Wird die Variable auf der rechten Seite eines mathematischen Ausdrucks eingesetzt, wird sie als Eingang festgelegt.

```
1  Def IO PORT1 = BIT,6
2  PORT1 = 1                         'Ausgang 6 wird eingeschaltet.
3  M1 = PORT1                        'Das Eingangssignal von Eingang 6 wird in M1
                                     eingelesen.
4  If PORT1 = 1 Then GoTo *Label     'Sonderfall!
                                     Eingang 6 wird auf log. 1 geprüft
```

△

6.3.25 Def Jnt (Define Joint)

Funktion: Gelenkvariable definieren

Dieser Befehl definiert eine Gelenkvariable. Er wird dann zur Variablendeklaration verwendet, wenn der Variablenname mit einem anderen Buchstaben als mit „J" beginnen soll. Variablennamen, die mit „J" beginnen, müssen nicht über den Befehl Def Jnt deklariert werden.

Eingabeformat

Def ☐ Jnt ☐ <Gelenkvariablenname> [,<Gelenkvariablenname>] ...

<Gelenkvariablenname> Legt den Namen der Gelenkvariablen fest

Programmbeispiel

```
1 Def Jnt SAFE                      'Deklariert SAFE als Namen einer Gelenkvariablen
2 MOV J1                            'Gelenkvariablen, deren Variablenname mit „J"
                                    beginnt, müssen nicht über den Befehl
                                    Def Jnt deklariert werden
3 SAFE = (–50,120,30,300,0,0,0,0)   'Weist der Variablen SAFE die angegebenen
                                    Werte zu
4 Mov SAFE                          'Fährt die Position SAFE an
```

Erläuterung

- Der Variablenname darf aus maximal 16 Zeichen bestehen. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 240 Zeichen (inklusive des Befehls) in einem Programmschritt verwendet werden.
- Durch einen Unterstrich „_" hinter dem „J" wird eine im Basisprogramm definierte Variable zur globalen Variablen und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.26 Def Plt (Define pallet)

Funktion: Palette definieren

Definiert eine Palette

Eingabeformat

Def ☐ Plt ☐ <Palettennummer>,<Bezugsposition>,<Endpunkt A>,<Endpunkt B>,
[<Palleteneckpunkt, der gegenüber der Bezugsposition liegt>],
<Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A>,
<Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B>,
<Bewegungsrichtung>

<Palettennummer>	Legt die Nummer der eingesetzten Palette fest 1 ≤ Palettennummer ≤ 8
<Bezugsposition>	Legt den Anfangspunkt der Palette fest
<Endpunkt A>	Legt einen der Endpunkte der Palette fest Dient als Zwischenposition für kreisförmige Paletten
<Endpunkt B>	Legt einen der Endpunkte der Palette fest Dient als Endpunkt für kreisförmige Paletten
<Palleteneckpunkt, der gegenüber der Bezugs-position liegt>	Legt den Punkt fest, der gegenüber der Bezugsposion liegt Für kreisförmige Paletten ohne Bedeutung
<Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A>	Legt die Gitterpunkte der Palette zwischen Bezugsposition und Endpunkt A fest Legt bei kreisförmiger Palette die Positionen zwischen dem Anfangs- und Endpunkt fest
<Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B>	Legt die Gitterpunkte der Palette zwischen Bezugsposition und Endpunkt B fest Für kreisförmige Paletten ohne Bedeutung (1 usw., muss festgelegt werden)
<Bewegungsrichtung>	Legt die Bewegungsrichtung und die Stellung fest. 1 = Zickzack (gleichmäßige Anpassung der Stellung) 2 = Bewegungsrichtung beibehalten (z. B. von links nach rechts) (gleichmäßige Anpassung der Stellung) 3 = kreisförmige Bewegung (gleichmäßige Anpassung der Stellung) 11 = Zickzack (feste Stellung) 12 = Bewegungsrichtung beibehalten (z. B. von links nach rechts) (feste Stellung) 13 = kreisförmige Bewegung (feste Stellung)

Programmbeispiel

```
1  Def Plt 1,P1,P2,P3, ,4,3,1        'Palettendefinition mit 3 Punkten
2  Def Plt 1,P1,P2,P3,P4,4,3,1       'Palettendefinition mit 4 Punkten
```

Palettendefinition und Bewegungsrichtung

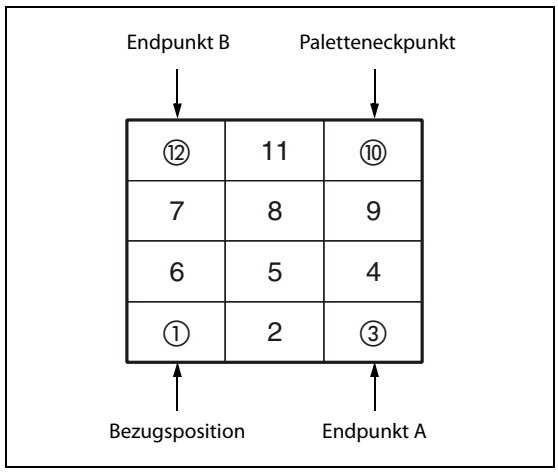


Abb. 6-10:
Palettendefinition mit Bewegungsrichtung = 1 (zickzack)

R000693C

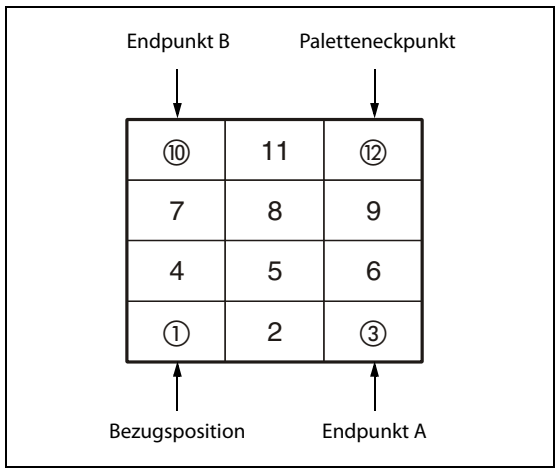


Abb. 6-11:
Palettendefinition mit Bewegungsrichtung = 2 (Richtung beibehalten)

R000694C

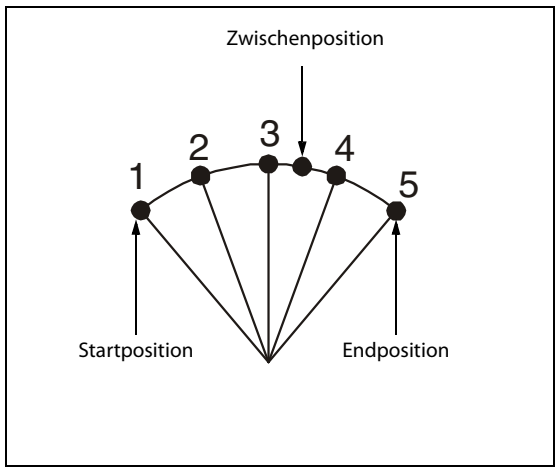


Abb. 6-12:
Palettendefinition mit Bewegungsrichtung = 3 (kreisförmig)

R000695C

Erläuterung

- Es können 3 oder 4 Punkte einer Palette definiert werden. Die Festlegung von 3 Punkten ist einfacher zu programmieren; durch die Festlegung von 4 Punkten erreicht man eine höhere Präzision.
- Der Befehl steht nur innerhalb des ausgeführten Programms zur Verfügung. Er kann nicht von einem anderen Programm aufgerufen werden. Falls nötig, muss er erneut definiert werden.
- Die Anzahl der Zeilen- und Spaltengitterpunkte muss größer als Null sein. Ansonsten erfolgt eine Fehlermeldung.
- Wenn das Produkt <Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A> × <Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B> den Wert 32767 überschreitet, erfolgt eine Fehlermeldung.
- Die Angabe der Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B ist für kreisförmige Paletten ohne Bedeutung, darf aber nicht weggelassen werden. Der Palletteneckpunkt, der gegenüber der Bezugsposition liegt, ist wirkungslos, auch wenn er festgelegt wurde. Setzen Sie „0“ oder einen formalen Wert.
- Bei nach unten gerichtetem Handflansch müssen die Vorzeichen der Stellungsdaten (A, B und C) an der Bezugsposition, dem Endpunkt A und dem Endpunkt B übereinstimmen. Bei einem Knickarm-Roboter mit nach unten gerichtetem Handflansch gilt: A = 180° (oder –180°), B = 0 und C = 180° (oder –180°). Stimmen die Vorzeichen der Stellungsdaten A und C an den 3 Positionen nicht überein, kann die Hand in der Mittelposition rotieren. Gleichen Sie in einem solchen Fall die Vorzeichen über die Teaching Box an. Dieselbe Position kann über einen Drehwinkel von +180° oder –180° erreicht werden. Eine Änderung der Vorzeichen dürfte somit problemlos durchzuführen sein.
- Bei einer Einstellung der Bewegungsrichtung auf einen Wert von 11 bis 13 wird die Stellung der Startposition während des gesamten Palettierungsvorgangs beibehalten. Ist die Bewegungsrichtung auf einen Wert von 1 bis 3 eingestellt, wird die Stellung entsprechend den Streckenabschnitten zwischen dem Start- und Endpunkt gleichmäßig angepasst.
- Bei einigen Robotermodellen (RV-2SQ usw.), deren J1- oder J4-Achse einen Drehwinkel von +/–180° überschreiten kann, ist die Definition einer Palette, bei der dieser Drehwinkel überschritten wird, nicht möglich. Wird eine solche Position definiert, erfolgt eine Fehlermeldung.
  Teilen Sie in einem solchen Fall die Palette bei der Definition auf (siehe Seite 4-29).

HINWEISE

Erreicht bei der Palettendefinition eine der Orientierungsdaten A, B oder C für die Einstellung der Bezugsposition, des Endpunkts A, des Endpunkts B oder des Palletteneckpunkts den Wert +/–180°, so kann die Hand bei unterschiedlichen Vorzeichen der Orientierungsdaten rotieren und sich in unvorhersehbarer Weise bewegen. Detaillierte Informationen dazu, wie Sie Orientierungsdaten verwenden, die +/–180° erreichen, finden Sie auf Seite 4-24.

Aus dem Wert für den Bezugspunkt der Palettendefinition werden durch den Palettierungsbefehl (Plt) die Stellungsmerker (FL1 der Positionsdaten) für die Gitterpunkte berechnet. Werden für die verschiedenen Punkte der Palettendefinition Positionsdaten mit unterschiedlichen Stellungsmerkern verwendet, so kann die Palettierung nicht ausgeführt werden.
Bei der Palettendefinition müssen die Stellungsdaten für die Bezugsposition, die Endpunkte A und B und den Palletteneckpunkt übereinstimmen. Aus der Bezugsposition werden auch die Werte die Multirotationsdaten für die Gitterpunkte abgeleitet (FL2 der Positionsdaten).
Werden für die verschiedenen Punkte der Palettendefinition Positionsdaten mit unterschiedlichen Stellungsmerkern für die Multirotationsdaten verwendet, so kann die Hand rotieren und sich in Abhängigkeit der Roboterstellungen bei der Palettierung und der Interpolationsbefehle (Gelenk-Interpolation, Linear-Interpolation usw.) in unvorhersehbarer Weise bewegen. Wählen Sie in einem solchen Fall die direkte oder indirekte Anfahrt mit Hilfe der Type-Funktion und stellen Sie so sicher, dass sich die Hand in der gewünschten Weise bewegt.

Steht in Beziehung zu folgenden Befehlen:

Plt

6.3.27 Def Pos (Define Position)

Funktion: Positionsvariable definieren

Dieser Befehl definiert eine XYZ-Positionsvariable. Er wird dann zur Variablendeklaration verwendet, wenn der Variablenname mit einem anderen Buchstaben als mit „P“ beginnen soll. Variablennamen, die mit „P“ beginnen, müssen nicht über den Befehl Def Pos deklariert werden.

Eingabeformat

Def ☐ Pos ☐ <Positionsvariablenname>[,<Positionsvariablenname>] ...

<Positionsvariablenname> Legt den Namen der Positionsvariablen fest

Programmbeispiel

```
1 Def Pos WORKSET                          'Deklariert WORKSET als Namen
                                           einer Positionsvariablen
2 Mov P1                                   'Positionsvariablen, deren Variablen-
                                           name mit „P“ beginnt, müssen nicht
                                           über den Befehl Def Pos deklariert
                                           werden
3 WORKSET = (250,460,100,0,0,–90,0,0)(0,0) 'Weist der Variablen WORKSET die
                                           angegebenen Werte zu
4 Mov WORKSET                              'Fährt die Position WORKSET an
```

Erläuterung

- Der Variablenname darf aus maximal 16 Zeichen bestehen. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 240 Zeichen (inklusive des Befehls) in einem Programmschritt verwendet werden.
- Durch einen Unterstrich „_“ hinter dem „P“ wird eine im Basisprogramm definierte Variable zur globalen Variablen und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.28 Dim (Dim)

Funktion: Dimension definieren

Legt die Anzahl der Elemente bei Feldvariablen fest. Es sind bis zu drei Dimensionen möglich.

Eingabeformat

Dim □ <Variablenname>(<max. Indexwert>[,<max. Indexwert>])
[,<Variablenname>(<max. Indexwert>[,<max. Indexwert>])] ...

<Variablenname>	Legt den Namen für die Feldvariable fest
<maximaler Indexwert>	Konstante, die die Anzahl der Elemente einer Feldvariablen festlegt 1 ≤ Maximaler Indexwert ≤ 999 Der maximale Indexwert darf nur Konstanten enthalten. Ausdrücke mit numerischen Operationen sind nicht erlaubt.

Programmbeispiel

```
1 Dim PDATA(10)              'Deklariert die Positions-Feldvariable PDATA als eine
                             Variable mit 10 Elementen
2 Dim MDATA#(5)              'Deklariert die Feldvariable MDATA# als eine Variable
                             mit 5 Elementen und doppelter Genauigkeit
3 Dim M1%(6)                 'Deklariert die Feldvariable M1% als eine Variable
                             vom Typ Integer mit 6 Elementen
4 Dim M2!(4)                 'Deklariert die Feldvariable M2! als eine Feldvariable
                             mit 4 Elementen und einfacher Genauigkeit
5 Dim CMOJI(7)               'Deklariert die Zeichenketten-Feldvariable CMOJI als
                             eine Variable mit 7 Elementen
6 Dim MD6(2,3), PD1(5,5)     'Deklariert die zweidimensionale numerische Feldvariable
                             MD6 vom Typ Real mit einfacher Genauigkeit als eine Variable
                             mit 2 × 3 Elementen
                             Deklariert die zweidimensionale Positions-Feldvariable
                             PD1 als eine Variable mit 5 × 5 Elementen
```

Erläuterung

- Es sind ein-, zwei- und dreidimensionale Felder erlaubt.
- Numerische Variablen sind in die Typen Integer und Real mit einfacher oder doppelter Genauigkeit aufgeteilt. Der Variablentyp wird über ein Symbol hinter dem Variablennamen festgelegt. Fehlt die Angabe des Variablentyps, wird der Typ Real mit einfacher Genauigkeit definiert. DimMABC(10) deklariert die Feldvariable MABC als eine Variable vom Typ Real mit einfacher Genauigkeit und 10 Elementen.
- Die Indexwerte für den Zugriff auf eine Feldvariable beginnen bei 1. Die Variable PDATA in Programmschritt 1des Programmbeispiels setzt sich aus den Elementen mit den Nummern 1 bis 10 zusammen.
- Der Wertebereich für die Indexwerte liegt zwischen 1 und 999. Ausdrücke mit numerischen Operationen sind nicht erlaubt. Der Systemspeicherbereich kann jedoch nicht überschritten werden.
  Ist die Anzahl der Elemente eine reelle Zahl, wird die Zahl automatisch auf eine Integer-Zahl gerundet. In Abhängigkeit des freien Systemspeicherplatzes ist die Anzahl der Elemente von Feldvariablen begrenzt. Erfolgt eine Fehlermeldung, ist die Speicherkapazität überschritten.
- Bei Überschreitung des Wertebereiches für den maximalen Indexwert erfolgt bei der Ausführung der Dim-Anweisung eine Fehlermeldung.
- Bei Ausführung der Dim-Anweisung sind die Feldvariablen auf keine Standardwerte gesetzt, sondern nicht definiert.
- Es können keine Felder ohne Dim-Anweisung verwendet werden.
- Die Dim-Anweisung steht nur innerhalb des ausgeführten Programmes zur Verfügung. Sie kann nicht von einem anderen Programm aufgerufen werden. Bei Verwendung in einem Unterprogramm muss die Dim-Anweisung erneut definiert werden.
- Feldvariablen können wie normale Variablen verwendet werden. Ein Variablennamen oder eine Zeichenkette zur Festlegung der Elementnummer mit mehr als 8 Zeichen kann jedoch nicht auf der Teaching Box angezeigt werden.
- Durch einen Unterstrich „_“ als zweites Zeichen im Variablennamen wird eine Variable als globale Variable gekennzeichnet und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.29 Dly (Delay)

Funktion: Verzögerung einstellen

Als einzelner Befehl wird zur festgelegten Zeit der Wartestatus erzeugt. Der Befehl dient zur Positionierung des Roboters und zur zeitabhängigen Steuerung von Ein- und Ausgangssignalen. Wird der Dly-Befehl für einen zusätzlichen Impulsausgang genutzt, wird die Impulsdauer festgelegt.

Eingabeformat

Als einzelner Befehl:

Dly ☐ <Zeit>

Für einen zusätzlichen Impulsausgang:

Beispiel: M_Out(1) = 1 DLY ☐ <Zeit>

<Zeit>	Legt die Dauer des Wartestatus oder die Impulsdauer in Sekunden fest Der kleinste einstellbare Wert ist 0,01 s. Auch die Einstellung 0,00 s ist zulässig. Der größte einstellbare Wert ist die größte Zahl vom Typ Real mit einfacher Genauigkeit.

Programmbeispiel

Wartestatus festlegen

```
1 Dly 30                    'Wartezeit von 30 Sekunden
```

Ausgabe eines Impulses

```
2 M_Out(17) = 1 Dly 0.5     'Setzt das allgemeine Ausgangssignal 17 für 0,5 s auf „1“
3 M_OutB(18) = 1 DLY 0.5    'Setzt das Signal 18 von den allgemeinen Ausgangssignalen
                            18 bis 25 für 0,5 s auf „1“ und die Signale 19 bis 25 nach 0,5 s auf „1“
```

Positioniervorgang abschließen

```
1 Mov P1                    'Position P1 mittels Gelenk-Interpolation anfahren
2 Dly 0.1                   'Der Abschluss des Positioniervorgangs erfolgt über die
                            Timer-Einstellung.
```

Handsteuerungsvorgang (Hand öffnen/schließen) abschließen

```
1 HOpen 1                   'Öffnet Hand 1
2 Dly 0.1                   'Um sicherzustellen, dass die Hand geöffnet ist, erfolgt
                            der Abschluss des Vorgangs über die Timer-Einstellung.
```

Erläuterung

- Der Dly-Befehl wird verwendet, um in Programmen Verzögerungszeiten zu erzeugen. Er dient ebenso zur zeitabhängigen Steuerung von Ein- und Ausgangssignalen, Bewegungsbefehlen und zur Festlegung einer Impulsdauer für ein Ausgangssignal in der OUT-Anweisung (siehe Programmschritt 2 im Programmbeispiel oben).
- Der Impulsausgang wird gleichzeitig mit Ausführung des im nächsten Programmschritt stehenden Befehls gesetzt.
- Es können bis zu 50 Impulsausgänge gleichzeitig gesteuert werden. Wird dieser Wert überschritten, kommt es bei Ausführung des Befehls zu einer Fehlermeldung.
- Bei der Steuerung eines Impulsausgangs wird jedes Bit nach Ablauf der eingestellten Zeit [z. B. M_Outb (8 Bits) oder M_Outw (16 Bits)] invertiert.
- Nach Ablauf der festgesetzten Zeit wird wieder der Zustand vor Ausführung des Befehls angenommen.
- Ein Programm endet ohne Berücksichtigung der eingestellten Zeitdauer für einen Impulsausgang mit Ausführung der End-Anweisung oder dem letzten Programmschritt. Nach Ablauf der Impulsdauer wird der Ausgang abgeschaltet.
- Die Reihenfolge der Prioritäten ist:
  COM > Act > WthIf (WTH) > Impulsausgang (Zeitintervall aktiv)
- Die Eingabe eines Stoppsignals während der Impulsausgabe führt nicht zur Unterbrechung der Impulsausgabe.

Beispiele▽

Bei Eingabe eines Stoppsignals in Programmschritt 2 wird die Ausführung des Programms unterbrochen, der Ausgangssignalzustand bleibt jedoch erhalten.

```
1 M_Out(17) = 1
2 Dly 10
3 M_Out(17) = 0
```

Bei der Steuerung eines Impulsausgangs wird jedes Bit nach Ablauf der eingestellten Zeit [z.B. M_Outb (8 Bits) oder M_Outw (16 Bits)] invertiert.

```
M_Outb(1) = 1 Dly 1.0
```

Das gezeigte Beispiel bewirkt die Ausgabe des Bitmusters 00000001 für eine Sekunde. Danach wird das Bitmuster 11111110 ausgegeben.

△

6.3.30 End (End)

Funktion: Programmende

Der Befehl definiert den letzten Programmschritt. Er kennzeichnet explizit das Ende eines Programms, so dass ein weiteres Unterprogramm nach der End-Anweisung angefügt wird.

Eingabeformat

End

Programmbeispiel

```
1  Mov P1                ’Position P1 mittels Gelenk-Interpolation anfahren
2  GoSub *ABC            ’Sprung zum Unterprogramm ABC
3  End                   ’Programmende
   :
10 *ABC                  ’Sprungmarke ABC festgelegt
11 M1 = 1                ’Numerische Variable M1 auf „1“ setzen
12 Return                ’Rücksprung zu Programmschritt 3
```

Erläuterung

- Der Befehl definiert den letzten Programmschritt. Soll ein Programm in der Mitte unterbrochen und in den Pausestatus gesetzt werden, verwenden Sie den Befehl Hlt.
- Bei einem über das Steuergerät gestarteten Programm erfolgt die Programmausführung kontinuierlich. Das Programm wird vom Beginn bis zur End-Anweisung und wieder von vorne ausgeführt. Eine Unterbrechung des Zyklus kann über die Taste END am Steuergerät erfolgen.
- Es können mehrere End-Anweisungen in einem Programm ausgeführt werden.
- Es muss nicht zwingend eine End-Anweisung an das Ende eines Programms gesetzt werden.
- Die Ausführung einer End-Anweisung in einem Unterprogramm, das durch einen CallP-Befehl aufgerufen wird, übergibt die Steuerung wieder an das Programm, in dem der CallP-Befehl ausgeführt wurde. Die Funktion entspricht der Return-Anweisung beim Befehl GoSub.
- Eine End-Anweisung im Hauptprogramm schließt alle mit dem Befehl Open geöffneten Dateien.
- Bei Ausführung der End-Anweisung werden die durch folgende Befehle eingestellten Werte zurückgesetzt: Spd, Accel, Oadl, JOvrd, Ovrd, Fine und Cnt.
- Das Ausführen der End-Anweisung führt nicht zwangsläufig zum Beenden des Programms. Durch das Umstellen des Parameters SLOTON auf den Wert 2 und Verändern des Parameters SLT1 von ..., REP, ..., ... auf ..., CYC, ..., ...bleibt der Roboter bei Erreichen von End stehen.

Steht in Beziehung zu folgenden Befehlen:

Hlt, CallP

6.3.31 Error (Error)

Funktion: Fehler generieren

Im Anwendungsprogramm wird ein Fehler erzeugt.

Eingabeformat

Error □ <Fehlernummer>

<Fehlernummer>	Es kann eine Konstante oder ein numerischer Ausdruck festgelegt werden. 9000 ≤ Fehlernummer ≤ 9299

Programmbeispiel

Fehler 9000 generieren

```
1  Error 9000                  'Generiert den Fehler mit der Fehlernummer 9000
```

In Abhängigkeit des Variablenwertes von M1 werden unterschiedliche Fehler generiert.

```
4  If M1 <> 0 Then *LERR       'Sprung zur Marke LERR, falls M1 ungleich 0 ist
   :
14 *LERR                       'Sprungmarke LERR festgelegt
15 MERR = 9000 + M1 * 10       'Berechnung der Fehlernummer in Abhängigkeit von M1
16 Error MERR                  'Generiert die berechnete Fehlernummer
17 End                         'Programmende
```

Erläuterung

- Bei schweren und leichten Fehlern erfolgt eine Programmunterbrechung. Die Programmschritte nach dem Error-Befehl werden nicht ausgeführt. Bei einer Warnung erfolgt keine Programmunterbrechung und das Programm wird mit dem nächsten Programmschritt weiter ausgeführt.
- Die Parameter UER1 bis UER20 ermöglichen die Definition von 20 Fehlermeldungen.
- Bei einer Fehlernummer außerhalb des gültigen Bereiches erfolgt die Meldung eines Systemfehlers.
- In Abhängigkeit der Fehlernummer reagiert das System wie in folgender Tabelle gezeigt:

Fehlernummer	Systemverhalten
9000–9099 (schwerer Fehler)	Die Programmausführung wird unterbrochen und die Servospannung abgeschaltet. Der Fehler wird bei einem Reset zurückgesetzt.
9100–9199 (leichter Fehler)	Die Programmausführung wird unterbrochen. Der Fehler wird bei einem Reset zurückgesetzt.
9200–9299 (Warnung)	Die Programmausführung wird nicht unterbrochen. Der Fehlerausgang wird bei einem Reset ausgeschaltet.

Tab. 6-8: *Systemverhalten in Abhängigkeit der Fehlernummer*

Steht in Beziehung zu folgenden Parametern:

UER1 bis UER20

6.3.32 Fine (Fine)

Funktion: Feinpositionierung

Legt den Status bei der Beendigung eines Interpolationsbefehls fest, wenn die Cnt-Einstellung gesperrt ist. In Abhängigkeit des Robotermodells (z. B. RH-Serie) kann eine Positionierung über den Befehl Dly effektiver sein als über den Befehl Fine.

Eingabeformat

Fine ☐ <Anzahl der Impulse> [,<Achse>]

<Anzahl der Impulse>	Festlegung der Anzahl der Impulse zur Positionierung Bei einer Einstellung von „0“ (Standardwert) ist die Feinpositionierung deaktiviert.
<Achse>	Festlegung der Achse, für die die Feinpositionierung ausgeführt werden soll Bei fehlender Angabe ist die Feinpositionierung für alle Achsen aktiviert.

Programmbeispiel

```
1 Fine 300          'Fine-Einstellung auf 300 Impulse festlegen
2 Mov P1            'Position P1 mittels Gelenk-Interpolation anfahren
3 Fine 100,2        'Feinpositionierung für Achse 2 auf 100 Impulse festlegen
4 Mov P2            'Position P2 mittels Gelenk-Interpolation anfahren
5 Fine 0            'Feinpositionierung deaktivieren
6 Mov P3            'Position P3 mittels Gelenk-Interpolation anfahren
7 Fine 100          'Feinpositionierung für alle Achsen 100 Impulse festlegen
8 Mov P4            'Position P4 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Beendigung von Bewegungsbefehlen wie z. B. Mov über die Fine-Anweisung findet nicht durch eine direkte Steuerung der Servomotoren statt. Es erfolgt vielmehr eine Prüfung, ob sich der Wert der Servorückmeldeimpulse innerhalb des gewünschten Bereiches befindet. Diese Methode ermöglicht eine genaue Positionierung des Roboters.
- Die Feinpositionierung über den Befehl Dly (Timer) ist einfacher zu definieren und kann in manchen Fällen bessere Ergebnisse als die Positionierung über den Fine-Befehl liefern.

Beispiel▽

```
1 Mov P1          'Position P1 mittels Gelenk-Interpolation anfahren
2 Dly 0.1         'Der Abschluss des Positioniervorgangs erfolgt über die
                  Timer-Einstellung.
```

△

- Während einer Programmabarbeitung ist die Fine-Einstellung solange gesperrt, bis sie durch das Programm freigegeben wird. Sobald die Fine-Einstellung freigegeben wurde, bleibt sie solange in diesem Zustand, bis sie erneut gesperrt wird.
- Nach Abarbeitung des Programms wird die Fine-Einstellung gesperrt (bei Ausführung der End-Anweisung oder bei Zurücksetzen des Programms während einer Programmunterbrechung).
- Ist die Cnt-Einstellung freigegeben, wird der Fine-Befehl ignoriert. Er wird auch dann ignoriert, wenn er freigegeben ist (d.h. er wird als gesperrt interpretiert, die Einstellung bleibt jedoch erhalten).
- Die Festlegung der Anzahl der Impulse ist für Zusatzachsen (zusätzliche Servoachsen) nicht möglich. Hier kann der Wert des Parameters „INP“ des Servoverstärkers verwendet werden. Der Befehl Fine wird bei Einstellungen des Servoverstärker-Parameters auf Integer-Werte ungleich 0 aktiviert und bei der Einstellung „0“ deaktiviert.
- Ist bei aktivierter Überschleiffunktion (Cnt 1) gleichzeitig über den Fine-Befehl eine Bedingung zum Abschluss des Interpolationsbefehls festgelegt, kann es vorkommen, dass der Roboter den Bewegungsbefehl nicht so weit ausführt, dass die Ausgabe des Impulses bei Erreichen der Zielposition erfolgt. Der Roboter wartet auf die Beendigung des Befehls und die Programmausführung wird unterbrochen. Aktivieren Sie die Überschleiffunktion nicht bei gleichzeitiger Verwendung der Feinpositionierung.

HINWEIS

Die Gelenkachsen der Roboter der RH-A/RH-S-Serie und der RV-SD/RH-SDH-Serie sind mit Encodern unterschiedlicher Auflösung (Anzahl der Impulse) ausgestattet. Ist der Wert der Fine-Anweisung „Anzahl der Impulse“ fest eingestellt, so brauchen die Roboter der RV-S/RH-S-Serie aufgrund der höheren Encoderauflösung länger, um eine Positionierung zu beenden. Werden die Roboter der RV-A/RH-A-Serie durch die Roboter der RV-SD/RH-SDH-Serie ersetzt, kann sich somit bei aktivierter Fine-Einstellung die Zeit bis zum Abschluss einer Positionierungsbewegung ändern.
Korrigieren Sie in diesem Fall die Abweichung durch eine Einstellung der „Anzahl der Impulse“ in der Fine-Anweisung.

6.3.33 Fine J (Fine Joint)

Funktion: Feinpositionierung bei Gelenk-Interpolation

Legt den Status bei der Beendigung eines Interpolationsbefehls durch eine Abweichung für eine Achse fest, wenn die Cnt-Einstellung gesperrt ist. Die Fine- und Fine-P-Anweisung sind bei aktiviertem Fine-J-Befehl deaktiviert.

Eingabeformat

Fine ☐ <Abweichung>, J [,<Achse>]

<Abweichung>	Die Abweichung wird durch eine Variable oder Konstante festgelegt. Bei einer Einstellung von „0“ (Standardwert) ist die Feinpositionierung deaktiviert. In Abhängigkeit des Koordinatensystems ist die Einheit „mm“ oder „Grad (deg)“. Der minimal einstellbare Wert ist „0,001“
<Achse>	Festlegung der Achse, für die die Feinpositionierung ausgeführt werden soll Bei fehlender Angabe ist die Feinpositionierung für alle Achsen aktiviert. Legen Sie die Achse als Konstante oder numerische Variable fest.

Programmbeispiel

```
1 Fine 1, J          'Fine-Einstellung für alle Achsen auf 1 [mm] (oder [Grad]) festlegen
2 Mov P1             'Position P1 mittels Gelenk-Interpolation anfahren
3 Fine 0.5, J, 2     'Feinpositionierung für Achse 2 auf 0,5 [mm] (oder [Grad]) festlegen
4 Mov P2             'Position P2 mittels Gelenk-Interpolation anfahren
5 Fine 0, J, 5       'Feinpositionierung für Achse 5 deaktivieren
6 Mov P3             'Position P3 mittels Gelenk-Interpolation anfahren
7 Fine 0, J          'Feinpositionierung für alle Achsen deaktivieren
8 Mov P4             'Position P4 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Der Fine-J-Befehl legt mit Hilfe von aus den Servorückmeldeimpulsen abgeleiteten Gelenkwerten fest, wie ein Bewegungsbefehl abgeschlossen wird (Positioniergenauigkeit). Diese Methode ermöglicht eine genaue Positionierung des Roboters.
- Die Verfahrbewegung ist abgeschlossen, wenn sich die Differenz zwischen dem Sollwert und den Servorückmeldeimpulsen für alle aktivierten Achsen innerhalb des durch die Abweichung festegelegten Bereiches befindet.
- Die Feinpositionierung über den Befehl Dly (Timer) ist einfacher zu definieren und kann in manchen Fällen bessere Ergebnisse als die Positionierung über den Fine-J-Befehl liefern.

Beispiel▽

```
1 Mov P1          'Position P1 mittels Gelenk-Interpolation anfahren
2 Dly 0.1         'Der Abschluss des Positioniervorgangs erfolgt über die
                  Timer-Einstellung.
```

△

- Während einer Programmabarbeitung ist die Fine-J-Einstellung solange gesperrt, bis sie durch das Programm freigegeben wird. Sobald die Fine-Einstellung freigegeben wurde, bleibt sie solange in diesem Zustand, bis sie erneut gesperrt wird.
- Nach Abarbeitung des Programms wird die Fine-Einstellung gesperrt (bei Ausführung der End-Anweisung oder bei Zurücksetzen des Programms während einer Programmunterbrechung).
- Ist die Cnt-Einstellung freigegeben, wird der Fine-J-Befehl ignoriert. Er wird auch dann ignoriert, wenn er freigegeben ist (d.h. er wird als gesperrt interpretiert, die Einstellung bleibt jedoch erhalten).
- Ist der Fine-J-Befehl aktiviert, sind die Befehle Fine und Fine P deaktiviert (die Einstellung bleibt nicht erhalten).
- Der Fine-J-Befehl kann auch bei Zusatzachsen verwendet werden.
- Ist bei aktivierter Überschleiffunktion (Cnt 1) gleichzeitig über den Fine-J-Befehl eine Bedingung zum Abschluss des Interpolationsbefehls festgelegt, kann es vorkommen, dass der Roboter den Bewegungsbefehl nicht so weit ausführt, dass die Ausgabe des Impulses bei Erreichen der Zielposition erfolgt. Der Roboter wartet auf die Beendigung des Befehls und die Programmausführung wird unterbrochen. Aktivieren Sie die Überschleiffunktion nicht bei gleichzeitiger Verwendung der Feinpositionierung.

6.3.34 Fine P (Fine Pause)

Funktion: Feinpositionierung durch geradlinigen Abstand

Legt den Status bei der Beendigung eines Interpolationsbefehls durch einen geradlinigen Abstand fest, wenn die Cnt-Einstellung gesperrt ist. Die Fine- und Fine-J-Anweisung sind bei aktiviertem Fine-P-Befehl deaktiviert.

Eingabeformat

Fine ☐ <Geradliniger Abstand>, P

<Geradliniger Abstand>	Festlegung in „mm“ durch eine Variable oder Konstante. Bei einer Einstellung von „0“ (Standardwert) ist die Feinpositionierung deaktiviert. Der minimal einstellbare Wert ist „0,001“

Programmbeispiel

```
1 Fine 1, P          'Fine-Einstellung auf 1 mm festlegen
2 Mov P1             'Position P1 mittels Gelenk-Interpolation anfahren
3 Fine 0, P          'Feinpositionierung deaktivieren
4 Mov P2             'Position P2 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Der Fine-P-Befehl legt mit Hilfe eines aus den Servorückmeldeimpulse abgeleiteten geradlinigen Abstands fest, wie ein Bewegungsbefehl abgeschlossen wird (Positioniergenauigkeit). Diese Methode ermöglicht eine genaue Positionierung des Roboters.
- Die Verfahrbewegung ist abgeschlossen, wenn sich die Differenz zwischen dem Sollwert und den Servorückmeldeimpulsen innerhalb des durch den geradlinigen Abstand festegelegten Bereiches befindet.
- Die Feinpositionierung über den Befehl Dly (Timer) ist einfacher zu definieren und kann in manchen Fällen bessere Ergebnisse als die Positionierung über den Fine-P-Befehl liefern.

Beispiel▽

```
1 Mov P1          'Position P1 mittels Gelenk-Interpolation anfahren
2 Dly 0.1         'Der Abschluss des Positioniervorgangs erfolgt über die
                  Timer-Einstellung.
```

△

- Während einer Programmabarbeitung ist die Fine-P-Einstellung solange gesperrt, bis sie durch das Programm freigegeben wird. Sobald die Fine-Einstellung freigegeben wurde, bleibt sie solange in diesem Zustand, bis sie erneut gesperrt wird.
- Nach Abarbeitung des Programms wird die Fine-Einstellung gesperrt (bei Ausführung der End-Anweisung oder bei Zurücksetzen des Programms während einer Programmunterbrechung).
- Ist die Cnt-Einstellung freigegeben, wird der Fine-P-Befehl ignoriert. Er wird auch dann ignoriert, wenn er freigegeben ist (d.h. er wird als gesperrt interpretiert, die Einstellung bleibt jedoch erhalten).
- Ist der Fine-P-Befehl aktiviert, sind die Befehle Fine und Fine J deaktiviert (die Einstellung bleibt nicht erhalten).
- Der Fine-P-Befehl kann nicht bei Zusatzachsen verwendet werden.
- Ist bei aktivierter Überschleiffunktion (Cnt 1) gleichzeitig über den Fine-P-Befehl eine Bedingung zum Abschluss des Interpolationsbefehls festgelegt, kann es vorkommen, dass der Roboter den Bewegungsbefehl nicht so weit ausführt, dass die Ausgabe des Impulses bei Erreichen der Zielposition erfolgt. Der Roboter wartet auf die Beendigung des Befehls und die Programmausführung wird unterbrochen. Aktivieren Sie die Überschleiffunktion nicht bei gleichzeitiger Verwendung der Feinpositionierung.

6.3.35 For-Next (For-Next)

Funktion: Programmschleife

Dieser Befehl bewirkt eine Wiederholung des Programmteils, der zwischen der For-Anweisung und der Next-Anweisung steht. Der Programmteil wird solange wiederholt, bis die Abbruchbedingungen erfüllt sind.

Eingabeformat

```
For □ <Zähler> = <Vorgabewert> To <Endwert>[Step<Schrittwert>]
 :
Next □ [<Zähler 1> [,<Zähler 2] ...]
```

<Zähler>	Der numerische Datentyp gibt die Anzahl der Wiederholungen der Programmschleife an. Gilt auch für Zähler 1 und Zähler 2 usw.
<Vorgabewert>	Gibt den Startwert des Zählers vor
<Endwert>	Gibt den Endwert des Zählers vor
<Schrittwert>	Legt die Schrittweite des Zählers fest Die Angabe des Wertes kann entfallen.

Programmbeispiel

Programm zur Addition der Zahlen 1 bis 10

```
1  MSUM = 0                'Weist MSUM den Wert 0 zu
2  For M1 = 1 To 10        'Erhöht den Zähler der numerischen Variablen
                           M1 von 1 bis 10 um 1
3  MSUM = MSUM + M1        'Addiert M1 zu der numerischen Variablen MSUM
4  Next M1                 'Sprung zum Programschritt 2
```

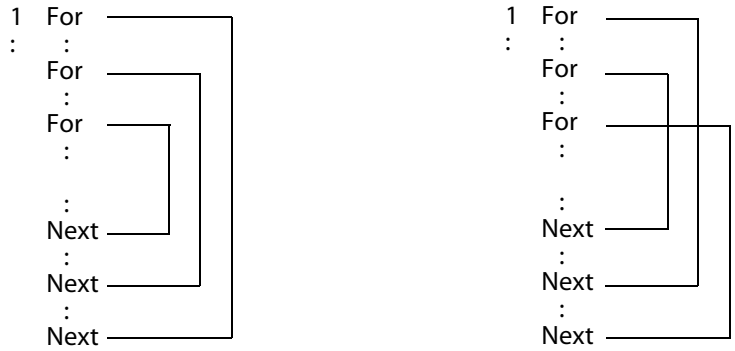
Speichert das Produkt zweier numerischer Variablen in eine zweidimensionale Feldvariable (Beispiel für verschachtelte For-Next-Programmschleifen)

```
1  Dim MBOX(10,10)         'Reserviert Speicherplatz für eine 10 × 10 Feldvariable
2  For M1 = 1 To 10 Step 1 'Erhöht den Zähler der numerischen Variablen M1 von 1
                           bis 10 um 1 und springt zum Programmschritt 7, sobald der
                           Wert 10 überschritten ist („STEP 1“ kann weggelassen werden).
3  For M2 = 1 To 10 Step 1 'Erhöht den Zähler der numerischen Variablen M2 von 1
                           bis 10 um 1 und springt zum Programmschritt 6, sobald der
                           Wert 10 überschritten ist („STEP 1“ kann weggelassen werden).
4  MBOX(M1,M2) = M1 * M2   'Ersetzt die Elemente der Feldvariablen MBOX(M1,M2)
                           durch das Produkt M1 * M2.
5  Next M2                 'Sprung zum Programmschritt 3
6  Next M1                 'Sprung zum Programmschritt 2
```

Erläuterung

- Es tritt ein Runtime-Fehler auf, wenn
  - der <Vorgabewert> größer als der <Endwert> und der <Schrittwert> positiv ist.
  - der <Vorgabewert> kleiner als der <Endwert> und der <Schrittwert> negativ ist.
- Erfolgt aus einer For-Next-Programmschleife ein Sprung über die GoTo-Anweisung, verringert sich der Speicherplatz (Stapelspeicher), der für die Programmstruktur zur Verfügung steht. Bei einer kontinuierlichen Ausführung des Programms kann eine Fehlermeldung auftreten. Bauen Sie das Programm so auf, dass die Schleife dann durchlaufen wird, wenn die For-Bedingung erfüllt ist.
- Werden For- und Next-Anweisungen nicht paarweise verwendet, tritt ein Runtime-Fehler auf.
- Steht die Next-Anweisung in unmittelbarer Beziehung zur nächsten For-Anweisung, können die Variablennamen in der Next-Anweisung weggelassen werden. „M2" in Programmschritt 5 und „M1" in Programmschritt 6 im Programmbeispiel können weggelassen werden. Dies führt zu einer höheren Verarbeitungsgeschwindigkeit.
- Programmebenen
  Es ist möglich, For-Next-Programmschleifen zwischen weiteren For-Next-Anweisungen zu verwenden. Mit jeder For-Next-Programmschleife erhöht sich die Zahl der Programmebenen um 1. Ein Programm darf aus maximal 16 Programmebenen bestehen. Bei mehr als 16 Ebenen erfolgt eine Fehlermeldung.
- Schleifen-Verschachtelungen

Beispiel▽



Richtig! **Falsch!**

△

6.3.36 FPrm (FPRM)

Funktion: Parameter definieren

Legt im Unterprogramm die Reihenfolge, den Typ und die Anzahl von Parametern fest, die von der CallP-Anweisung eines Hauptprogramms übergeben werden.

Eingabeformat

FPrm ☐ <Formalparameter>[,<Formalparameter>] ...

<Formalparameter>	Variablenname im Unterprogramm Es können alle Variablen verwendet werden. Es dürfen maximal 16 Variablentypen verwendet werden.

Programmbeispiel

```
1  FPrm M1,P1,P2        'Festlegung der Datentypen, der Reihenfolge und
                        der Anzahl
```

Wenn das Unterprogramm den Namen 20 trägt, so ist die Übergabe der Variablen M4, P3 und P5 vom Hauptprogramm an das Unterprogramm in die Variablen M100, P1 und P30 wie folgt zu definieren:

Hauptprogramm:

```
 :
12  CallP "P20", M4, P3, P5
 :
```

Unterprogramm:

```
1  FPrm M100, P1, P30
```

Die Variableninhalte von M4 wird an M100, P3 an P1 und P5 an P30 übergeben.

Erläuterung

- Der FPrm-Befehl wird nicht benötigt, wenn im aufgerufenen Unterprogramm keine Parameter verwendet werden.
- Weicht die Anzahl oder der Typ der in der CallP-Anweisung aufgeführten Formalparameter von denen im FPrm-Befehl ab, erfolgt eine Fehlermeldung.
- Rechenergebnisse eines Unterprogramms können nicht mittels temporärer Parameter in das Hauptprogramm übertragen werden. Verwenden Sie zu diesem Zweck externe Variablen.

Steht in Beziehung zu folgenden Befehlen:

CallP

6.3.37 GetM (Get Mechanism)

Funktion: Mechanismus definieren

Dieser Befehl definiert einen Mechanismus oder zusätzliche Achsen für den Multitasking-Betrieb. Die Zuordnung kann über den RelM-Befehl aufgehoben werden.

Eingabeformat

GetM □ <Mechanismusnummer>

<Mechanismusnummer>	Festlegung der Mechanismusnummer über einen numerischen Wert oder eine Variable 1 ≤ Mechanismusnummer ≤ 3 Der Roboterarm in einem Standardsystem wird als Mechanismus 1 definiert.

Programmbeispiel

Programmplatz 2 wird über den Programmplatz 1 gestartet. Der Mechanismus 1 im Programmplatz 2 wird über Programmplatz 1 gesteuert.

```
1  RelM                  'Definition von Mechanismus 1 aufheben, um
                         Mechanismus 1 über Programmplatz 2 zu steuern
2  XRun 2,"10"           'Startet Programm 10 als Programmplatz 2
30 Wait M_Run(2) = 1     'Wartestatus, bis Betriebssignal des Programmplatzes 2
                         gleich 1 ist
   :
```

Programmplatz 2 (Programm 10)

```
1  GetM 1                'Definition des Mechanismus 1
2  Servo On              'Servospannung des Mechanismus 1 einschalten
3  Mov P1                'Position 1 mittels Gelenk-Interpolation anfahren
4  Mvs P2                'Position 2 mittels Linear-Interpolation anfahren
5  P3 = P_Curr           'Aktuelle Position von Mechanismus 1 in P3 übertragen
6  Servo Off             'Servospannung des Mechanismus 1 ausschalten
7  RelM                  'Definition von Mechanismus 1 aufheben
8  End                   'Programmende
```

Erläuterung

- In der Regel, d. h. beim Einzelprogrammplatzbetrieb, wird der Befehl GetM nicht benötigt, da der Betrieb des Mechanismus 1 standardmäßig vordefiniert ist.
- Ein Mechanismus oder eine Zusatzachse kann nicht gleichzeitig über mehrere Programmplätze angesteuert werden.
- Wird beim Multitasking ein Mechanismus oder eine Zusatzachse nicht im Programmplatz 1 betrieben, muss nach Aufhebung der Mechanismusdefinition über den Befehl RelM in Programmplatz 1 eine Zuordnung über den Befehl GetM im Robotersteuerprogramm erfolgen. Es erfolgt eine Fehlermeldung, wenn der GetM-Befehl für einen Mechanismus ausgeführt wird, der bereits über den Befehl definiert worden ist.
- Befehle zum Ein- und Ausschalten der Servomotoren, Interpolationsbefehle, Befehle zur Festlegung von Beschleunigungs- und Bremszeiten sowie die Befehle Tool und Base können nur nach Definition eines Mechanismus über den GetM-Befehl ausgeführt werden.
- Bei fehlender Angabe des Arguments werden den Roboterstatusvariablen, die eine Mechanismusdefinition benötigen, die Werte des aktuellen Mechanismus zugewiesen.
- Bei einem Programmstopp wird der Befehl RelM automatisch durch das System ausgeführt. Bei einem Neustart des Programms wird der Befehl GetM automatisch ausgeführt.
- Der Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

Steht in Beziehung zu folgenden Befehlen:

RelM

6.3.38 GoSub (Go Subroutine)

Funktion: Sprung zu einem Unterprogramm

Bewirkt einen Sprung zu einem Unterprogramm, das mit einer Marke beginnt. Der Rücksprung muss über die Return-Anweisung erfolgen.

Eingabeformat

GoSub □ <Sprungziel>

<Sprungziel>	Legt eine Marke fest

Programmbeispiel

```
1  GoSub *LBL          'Sprung zum Unterprogramm LBL
2  End                 'Programmende
   :
20 *LBL                'Sprungmarke LBL festgelegt
21 Mov P1              'Position P1 anfahren
22 Return              'Rücksprung zum Programmschritt 2
```

Erläuterung

- Der Rücksprung aus dem Unterprogramm muss mit der Return-Anweisung erfolgen. Ein Rücksprung über die GoTo-Anweisung verringert den für die Programmsteuerung zur Verfügung stehenden Speicherplatz (Stapelspeicher) und führt bei kontinuierlicher Ausführung zu einer Fehlermeldung.
- Aus einem Unterprogramm kann über die GoSub-Anweisung der Sprung zu weiteren Unterprogrammen erfolgen. Dabei ist eine Verschachtelungstiefe von ca. 800 Sprüngen möglich.
- Ist das angegebene Sprungziel nicht vorhanden, erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Befehlen:

Return

6.3.39 GoTo (Go To)

Funktion: Sprung zu einer Marke

Bewirkt einen unbedingten Sprung zu einer Marke.

Eingabeformat

GoTo □ <Sprungziel>

<Sprungziel>	Legt eine Marke fest

Programmbeispiel

```
      :
1     GoTo *LBL          'Sprung zur Marke LBL
      :
100   *LBL               'Sprungmarke LBL festgelegt
101   Mov P1             'Position P1 anfahren
```

Erläuterung

- Ist das angegebene Sprungziel nicht vorhanden, erfolgt eine Fehlermeldung.

6.3.40 Hlt (Halt)

Funktion: Programmablauf stoppen

Der Befehl stoppt das Programm, in dem er ausgeführt wurde. Auf andere Programme, die im Rahmen eines Multitaskings parallel ausgeführt werden, hat der Befehl keinen Einfluss.

Eingabeformat

Hlt

Programmbeispiel

Der Roboter wird während der Programmausführung unter Angabe einer Bedingung gestoppt.

```
10 If M_In(18) = 1 Then Hlt          'Stoppt den Roboter beim Einschalten des
                                     Eingangssignals 18
11 Mov P1 WthIf M_In(17) = 1,Hlt     'Position 1 anfahren und Programm
                                     stoppen, falls das Eingangsbit 17 gleich 1 ist
```

Der Roboter wird während der Programmausführung ohne Angabe einer Bedingung gestoppt.

```
15 HLT                               'Stoppt den Roboter ohne Bedingung
```

Erläuterung

- Der Befehl unterbricht den Programmablauf des Programms, in dem er ausgeführt wurde und stoppt den Roboter mit der definierten Abbremszeit.
- Bei Ausführung des Hlt-Befehls im Multitasking-Betrieb wird nur das Programm in dem Programmplatz unterbrochen, in dem der Hlt-Befehl ausgeführt worden ist.
- Ein Neustart kann über die Teaching Box oder durch ein externes Start-Signal erfolgen. Der Programmstart beginnt einen Programmschritt nach dem Hlt-Befehl. Wurde der Hlt-Befehl in einer Verknüpfung ausgeführt, startet das Programm bei dem Programmschritt, bei dem es unterbrochen wurde.

ACHTUNG:

Bei Verwendung der Funktion zur Bandverfolgung:

- ***ab Software-Version S1 (SD-Serie), R1 (SQ-Serie)***
  Bei Ausführung des Befehls während einer Bewegung zur Bandverfolgung, wird die Bewegung gestoppt (wie bei Ausführung des Trk-Off-Befehls) und die Programmausführung unterbrochen. Bei Multimechanismen wird die Bewegung des Roboters gestoppt, dem die Mechanismusnummer mit dem GetM-Befehl zugewiesen wurde. Soll die Bewegung bei einem erneuten Start fortgesetzt werden, verwenden Sie im Programm den Trk-On-Befehl.
- ***vor Software-Version S1 (SD-Serie), R1 (SQ-Serie)***
  Bei Ausführung des Befehls während einer Bewegung zur Bandverfolgung, wird die Programmausführung unterbrochen, die Bewegung aber weiter ausgeführt. Soll die Bewegung gestoppt werden, führen Sie vor der Ausführung des Hlt-Befehls den Trk-Off-Befehl aus.

Steht in Beziehung zu folgenden Befehlen:

End

6.3.41 HOpen/HClose (Hand Open/Hand Close)

Funktion: Handgreiferzustand festlegen

Legt den Handgreiferzustand (offen/geschlossen) fest.

Eingabeformat

```
HOpen □ <Handnummer>
HClose □ <Handnummer>
```

<Handnummer>	Festlegung der Handnummer als Konstante oder Variable 1 ≤ Handnummer ≤ 8

Programmbeispiel

```
1 HOpen 1          'Öffnet Hand 1
2 Dly 0.2          'Wartezeit von 0,2 Sekunden ermöglicht ein sicheres
                   Öffnen der Hand
3 HClose 1         'Schließt Hand 1
4 Dly 0.2          'Wartezeit von 0,2 Sekunden ermöglicht ein sicheres
                   Schließen der Hand
5 Mov PUP          'Position PUP anfahren
```

Erläuterung

- Die Funktion der Magnetventile (einfach/doppelt) wird in Parameter HANDTYPE festgelegt.
- Bei Einstellung der Funktion auf „doppelt" können Hand 1 bis Hand 4, bei Einstellung der Funktion auf „einfach" können Hand 1 bis Hand 8 gesteuert werden.
- Das Handsteuersignal (auf/zu) nach Einschalten der Versorgungsspannung kann über Parameter HANDINIT festgelegt werden.
- Die Roboterstatusvariable M_HndCq ermöglicht die Abfrage des Handgreiferstatus. Auf das Signal kann auch über die Eingangssignale Nr. 900 bis 907 (bei Definition eines Mechanismus) zugegriffen werden.

```
1 HCLOSE 1                    'Schließt Hand 1
2 *REP                        'Sprungmarke REP festgelegt
3 If M_HndCq(1) <> 1 GoTo *REP 'Wartestatus, bis die Roboterstatusvariable
                               M_HNDCQ(1) gleich 1 ist
4 Mov P1                      'Position 1 mittels Gelenk-Interpolation anfahren
```

- Auf den Handgreifer bezogene Parameter finden Sie im Kap. 9 „Parameter".

Steht in Beziehung zu folgenden Systemvariablen:

M_In/M_Inb/M_In8/M_Inw/M_In16 (ab Signalnummer 900),
M_Out/M_Outb/M_Out8/M_Outw/M_Out16 (ab Signalnummer 900), M_HndCq

Steht in Beziehung zu folgenden Befehlen:

LoadSet, Mxt

Steht in Beziehung zu folgenden Parametern:

HANDTYPE, HANDINIT

6.3.42 If ... Then ... Else (If Then Else)

Funktion: WENN ... DANN ... SONST

WENN eine bestimmte Bedingung zutrifft, DANN führe Anweisung 1 aus, SONST führe Anweisung 2 aus.

Eingabeformat

```
If ☐ <Ausdruck> ☐ Then ☐ <Anweisung> ☐ [Else <Anweisung>]
```

```
If ☐  <Ausdruck> ☐ Then
      <Anweisung>
      <Anweisung>
      Break
        :
[Else]
      <Anweisung>
      <Anweisung>
      Break
        :
EndIf
```

<Ausdruck>	Beschreibt einen booleschen Ausdruck
<Anweisung>	Anweisung nach Then wird ausgeführt, wenn das Ergebnis des booleschen Ausdrucks „wahr“ ist. Anweisung nach Else wird ausgeführt, wenn das Ergebnis des booleschen Ausdrucks „unwahr“ ist.

Programmbeispiel

```
1   If M1 > 10 Then *L1                    'Sprung zur Marke L1, falls M1 größer 10
11  If M1 > 10 Then GoTo *L2 Else GoTo *L3 'Sprung zur Marke L2, falls M1 größer 10,
                                           sonst Sprung zur Marke L3
                                           Die Anweisung GoTo nach Then oder Else kann
                                           entfallen
 :
19  *L1                                    'Sprungmarke L1 festgelegt
20  M1 = 10                                'Setzt M1 auf den Wert „10“
21  Mov P1                                 'Position P1 anfahren
22  GoTo *LC                               'Sprung zur Marke LC
23  *L2                                    'Sprungmarke L2 festgelegt
24  M1 = –10                               'Setzt M1 auf den Wert „–10“
25  Mov P2                                 'Position P2 anfahren
26  GoTo *LC                               'Sprung zur Marke LC
```

Erläuterung

- Die If ... Then ... Else-Anweisungen müssen in einem Programmschritt aufgeführt sein.
- Ein If ... Then ... Else ... Endlf-Programmblock kann in mehrere Programmschritte aufgeteilt werden.
- Die Else-Anweisung kann entfallen.
- Ein If ... Then ... Else ... Endlf-Programmblock muss die Anweisung Endlf enthalten.
- Ein Rücksprung aus einem If ... Then ... Else ... Endlf-Programmblock über die GoTo-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.
- Innerhalb eines If ... Then ... Else ... Endlf-Programmblocks können zwischen den Anweisungen If und Else weitere If ... Then ... Else ... Endlf-Programmblöcke ausgeführt werden. Dabei ist eine Verschachtelungstiefe von bis zu 8 Programmebenen möglich.
- Nach einer Then- oder Else-Anweisung kann die GoTo-Anweisung entfallen.
  Beispiel: If M1 > 10 Then *L200 Else *L300
  Folgt eine GoTo-Anweisung auf eine Then-Anweisung, kann entweder die GoTo- oder die Then-Anweisung entfallen.
  Beispiel: If M1 > 10 Then GoTo *L200 entspricht der Programmzeile
  If M1 > 10 Then *L200 oder der Programmzeile
  If M1 > 10 GoTo *L200
- Nach einer Then- oder Else-Anweisung ermöglicht die Break-Anweisung einen Sprung zu dem Programmschritt, der auf die Endlf-Anweisung folgt.

6.3.43 Input # (Input)

Funktion: Eingabe

Mit dieser Anweisung können Daten aus Dateien oder Eingabegeräten im ASCII-Format eingelesen werden. Hinweise zu den entsprechenden Parametern finden Sie im Abschn. 9.16.

Eingabeformat

Input □ # <Dateinummer>,<Datenname>[,<Datenname>] ...

<Dateinummer>	Legt die Dateinummer fest 1 ≤ Dateinummer ≤ 8
<Datenname>	Name der Variablen, in die die Daten übertragen werden Es können alle Variablentypen verwendet werden.

Programmbeispiel

```
1  Open "COM1:" AS #1      '„COM1:“ wird als Datei Nummer 1 geöffnet
2  Input #1,M1             'Erfolgt eine Eingabe von der Tastatur, wird dieser Wert
                           in die numerische Variable M1 übertragen.
3  Input #1,CABC$          'Erfolgt eine Eingabe von der Tastatur, wird dieser Wert
                           in die Zeichenkettenvariable CABC$ übertragen.
   :
10 Close #1                'Datei Nummer 1 schließen
```

Erläuterung

- Überträgt Eingangsdaten aus Dateien (oder von Eingabegeräten), die mittels Open-Anweisung geöffnet worden sind, in eine Variable. Ist die Open-Anweisung nicht ausgeführt worden, erfolgt eine Fehlermeldung.
- Der übertragene Datentyp und der Variablentyp müssen übereinstimmen.
- Werden mehrere Variablennamen angegeben, müssen sie durch Kommas getrennt werden.
- Bei Ausführung der Input-Anweisung wartet das System auf eine Eingabe. Bei Betätigung der Eingabetaste (CR und LF) werden die Eingangsdaten in die Variablen übertragen.
- Beim Senden von Werten an den Roboter ist vor den Daten die Buchstabenfolge PRN zu setzen. Beispiel: PRN 50 für die Übergabe des Wertes 50
- Bei Eingabe mehrerer Elemente werden die Elemente der Reihe nach übertragen.
  Beispiel:
  Bei Eingabe einer Zeichenkette, eines numerischen Wertes und einer Position
  1 Input #1,C1$,M1,P1
  PRN MELFA,125.75,(130.5,–117.2,55.1,16.2,0,0)(1,0) CR
  „MELFA“ wird in C1$ übertragen,
  125.75 in M1 und
  (130.5,–117.2,55.1,16.2,0,0)(1,0) in P1
  Ist die Anzahl der Elemente größer als in der Input-Anweisung angegeben, werden die überzähligen Elemente nicht eingelesen.

Steht in Beziehung zu folgenden Befehlen:

Open, Close, Print

6.3.44 JOvrd (J Override)

Funktion: Übersteuerung

Legt die Geschwindigkeit für die Gelenk-Interpolation fest

Eingabeformat

JOvrd □ <Übersteuerungswert>

<Übersteuerungswert>	Legt den prozentualen Übersteuerungswert als reelle Zahl oder als numerischen Ausdruck fest 1 ≤ Übersteuerungswert ≤ 100.0

Programmbeispiel

```
1 JOvrd 50          'Übersteuerung auf den Wert 50 % einstellen
2 Mov P1            'Position P1 anfahren
3 JOvrd M_NJovrd    'Standardwert einstellen
```

Erläuterung

- Der JOvrd-Befehl ist nur bei der Gelenk-Interpolation wirksam.
- Die aktuelle Arbeitsgeschwindigkeit ergibt sich folgendermaßen:

Gelenk-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des Ovrd-Befehls	×	Einstellwert des JOvrd-Befehls

- Der Maximalwert der Arbeitsgeschwindigkeit ist 100 %. Der Standardwert der Arbeitsgeschwindigkeit beträgt 100 % der Standardeinstellung (M_NOvrd).
- Die Arbeitsgeschwindigkeit wird bei Ausführung der End-Anweisung und bei einem Reset auf den Standardwert zurückgesetzt.

Steht in Beziehung zu folgenden Systemvariablen:

M_JOvrd/M_NJovrd/M_OPovrd/M_Ovrd/M_NOvrd
M_NJovrd (Standardeinstellung), M_JOvrd (aktuelle Einstellung)

Steht in Beziehung zu folgenden Befehlen:

Ovrd, Spd

6.3.45 JRC (Joint Roll Change)

Funktion: Gelenkposition verändern

Roboterarmachse:
Überschreibt die aktuelle Position durch Addition von ±360° zur aktuellen Gelenkposition der entsprechenden Achse (siehe Eingabeformat unter „Achsennummer" unten).

Benutzerdefinierte Achse:
Überschreibt die aktuelle Position durch Addition oder Subtraktion eines festgelegten Wertes zur aktuellen Gelenkposition der gewählten Achse. Der Anwender kann den Wert in einem Parameter festlegen. Als Achse kann sowohl eine Gelenkachse als auch eine lineare Achse gewählt werden.
Die Grundposition eines Gelenkes kann verändert werden.

Eingabeformat

JRC □ <[+]1/–1/0>[,<Achsennummer>]

JRC □ <[+]<Numerischer Wert>/-<Numerischer Wert>/0>[,<Achsennummer>]

<+1>	Addiert einen definierten Wert zur aktuellen Position einer gewählten Gelenkachse Der Wert, um den die Position verändert wird, ist im Parameter JRCQTT festgelegt. Bei der Standardachse ist dieser Wert auf 360° festgelegt.
<–1>	Subtrahiert einen definierten Wert von der aktuellen Position einer gewählte Gelenkachse Der Wert, um den die Position verändert wird, ist im Parameter JRCQTT festgelegt. Bei der Standardachse ist dieser Wert auf 360° festgelegt.
<0>	Der aktuelle Wert einer festgelegten Achse wird als Grundposition in den Parameter JRCORG geschrieben. Dieser Befehl kann nur bei einer benutzerdefinierten Achse ausgeführt werden.
<Achsennummer>	Die Zielachse wird durch Jx angegeben, wobei x eine Zahl zwischen 1 und 8 ist. Erfolgt keine Angabe, werden die Standardachsen verwendet. Bei Zusatzachsen oder benutzerdefinierten Zusatzachsen darf die Angabe nicht wegelassen werden. Robotertypen und Standardachsen: RV-SD: J6-Achse RH-SDH: J4-Achse Es können bei allen Robotertypen benutzerdefinierte Zusatzachsen als Zielachse definiert werden. Es können alle Achsen benutzerdefinierter Mechanismen als Zielachse definiert werden.
<Numerischer Wert>	Legt den Wert ganzer Umdrehungen fest, die zu der aktuellen Position einer Gelenkachse addiert werden Beispiel: +3: Zunahme des Winkels der Zielachse um 1080° –2: Abnahme des Winkels der Zielachse um 720°

Programmbeispiel

```
1 Mov P1          'Position 1 anfahren (Ausgangsposition)
2 JRC +1          'Addition von 360° zur aktuellen Position der J6-Achse
3 Mov P1          'Position 1 anfahren
4 JRC +1          'Addition von 360° zur aktuellen Position der J6-Achse
5 Mov P1          'Position 1 anfahren
6 JRC –2          'Subtraktion von 720° von der aktuellen Position der
                  J6-Achse (Rückkehr zur Ausgangsposition)
```

Erläuterung

- JRC = 1/–1 (JRC n/ –n) erhöht/verringert den aktuellen Gelenkwinkel um einen festgelegten Wert.
- Über JRC = 0 wird die Grundposition einer festgelegten Achse mit der aktuellen Position überschrieben.
- Verwenden Sie den JRC-Befehl, muss im Voraus der Bewegungsbereich der Zielachse verändert werden. Dies ist notwendig, damit der Roboter bei Ausführung des Befehls den Bewegungsbereich nicht verlässt. Im Parameter MEJAR verändern Sie dazu die Minus und Pluswerte. Der Bewegungsbereich der Drehachsen liegt zwischen –2340° und 2340°.
- Wird keine Achse festgelegt, wird als Zielachse automatisch die Standardachse eingestellt. Dies ist bei allen Robotern die letzte Rotationsachse.
- Existiert die festgelegte Achse bei dem verwendeten Roboter nicht, oder ist sie im Sinne des JRC-Befehls keine Zielachse, wird bei Ausführung des JRC-Befehls ein Fehler angezeigt.
- Ist keine Grundposition definiert, erfolgt eine Fehlermeldung, sobald der JRC-Befehl ausgeführt wird.
- Aufgrund des JRC-Befehls stoppt der Roboter. Bei Freigabe des Cnt-Befehls und Verwendung des JRC-Befehls ist die Bewegung nicht mehr interpoliert.
- Bevor Sie den JRC-Befehl verwenden, müssen die folgenden Parameter eingestellt sein:
  - JRCEXE = 1 (JRC-Ausführung freigegeben)
  - Änderung des Bewegungsbereiches der Zielachse über MEJAR
  - Einstellung des Wertes in JRCQTT (nur bei benutzerdefinierten Achsen möglich)
  - Einstellung der Grundposition über JRCORG
- Bei JRCEXE = 0 kann der JRC-Befehl nicht ausgeführt werden.
- Die Bewegung ist über den Parameter JRCQTT festgelegt. Liegt dieser Parameter nicht innerhalb der Pulsdaten 0–MAX, wird bei der Initialisierung ein Fehler angezeigt. Der MAX-Wert ist definiert durch: MAX = 2^(Anzahl der Encoder-Bits + 15) – 1.

Beispiel▽

Ein 13-Bit-Encoder (8192 Pulse): MAX = 2^(13 + 15) – 1
MAX = 0x0fffffff
Ein 14-Bit-Encoder (16384 Pulse): MAX = 2^(14 + 15) – 1
MAX = 0x1fffffff

△

Pulsdaten werden in Bewegungsdaten konvertiert.

Für Drehachsen:
Pulsdaten = Bewegung(°/360) × Anzahl der Encoder-Pulse

Für lineare Achsen:
Pulsdaten = Bewegung (mm/360) × Anzahl der Encoder-Pulse

- Muss das Steuergerät während eines Updates initialisiert werden, speichern Sie vorher die Parameter im Grundzustand.
- Das schrittweise Ausführen (rückwärts) des Programms ist mit dem JRC-Befehl nicht möglich.
- Die Anweisung kann in einem kontinuierlich ausgeführtem Programm nicht verwendet werden.

Steht in Beziehung zu folgenden Parametern:

JRCEXE	Freigabe des JRC-Befehls (freigegeben/gesperrt = 1/0) Standardeinstellung JRCEXE = 0
JRCQTT	Festlegung der Werte, um die die festgelegten Achsen mit dem JRC-Befehl verschoben werden. Dieser Wert ist bei der Standardachse auf 360° festgelegt.
JRCORG	Festlegung der Grundposition für JRC = 0 (kann nur bei zusätzlichen und benutzerdefinierten Achsen eingestellt werden)

6.3.46 Label (Label)

Funktion: Sprungmarke

Legt ein Sprungziel fest

Eingabeformat

*<Name der Marke>

*<Name der Marke>[:<Befehlszeile>]

<Name der Marke>	Legt den Namen der Marke über eine Zeichenkette fest Das erste Zeichen muss ein Buchstabe sein. Die maximale Länge beträgt 16 Zeichen (Das (*)-Zeichen wird nicht mitgezählt.)
<Befehlszeile>	Legt eine Befehlszeile hinter dem Markennamen nach dem Doppelpunkt fest

Programmbeispiel

```
1  *SUB1                                    'Sprungmarke SUB1 festgelegt
2   If M1 = 1 Then GoTo *SUB1               'Sprung zum Unterprogramm SUB1,
                                            wenn M1 gleich 1
3   *LBL1: If M_In(19) = 0 Then GoTo *LBL1  'Durchläuft die Warteschleife in
                                            Programmschritt 3 bis M_In(19) auf „1“
                                            gesetzt wird.
```

Erläuterung

- Es erfolgt keine Fehlermeldung, wenn die Marke während eines Programmablaufes nicht aufgerufen wird.
- Ist die gleiche Marke in einem Programm mehrmals definiert, erfolgt eine Fehlermeldung.
- Reservierte Wörter dürfen nicht als Markennamen vergeben werden.
- Im Markennamen darf an der zweiten Stelle kein Unterstrich („_“) verwendet werden. Diese Schreibweise ist den externen Variablen vorbehalten. Wird an der zweiten Stelle ein Unterstrich verwendet, z. B. „*A_“, „*B_“ oder „*Z_“, erfolgt eine Fehlermeldung. Auch ein Markenname wie „*L_“ erzeugt bei der Ausführung des Programms eine Fehlermeldung. Die Schreibweise ist für das System reserviert.
  Der Unterstrich kann in Markennamen ab der dritten Stelle verwendet werden.
- Auf die Markendefinition kann nach einem Doppelpunkt eine Befehlszeile folgen. Das Anhängen weiterer Befehle ist in dieser Zeile jedoch nicht möglich.

6.3.47 LoadSet (Load Set)

Funktion: Hand- und Werkstückbedingung einstellen

Legt die Hand- und Werkstückbedingungen für eine optimale Beschleunigung/Abbremsung (Oadl) fest

Eingabeformat

LoadSet □ <Handnummer>,<Werkstücknummer>

<Handnummer>	Legt die Hand für die Einstellung der Größe und des Gewichts für die optimale Beschleunigung/Abbremsung fest (1 ≤ HNDDAT ≤ 8)
<Werkstücknummer>	Legt das Werkstück für die Einstellung der Größe und des Gewichts für die optimale Beschleunigung/Abbremsung fest (1 ≤ WRKDAT ≤ 8)

Programmbeispiel

```
1 LoadSet 1,1     'Optimale Beschleunigung/Abbremsung für Hand 1
                  (HNDDAT1) und Werkstück 1 (WRKDAT1) einstellen
2 Oadl On         'Optimale Beschleunigung/Abbremsung freigeben
3 Mov P1          'Position 1 mittels Gelenk-Interpolation und optimaler
                  Beschleunigung/Abbremsung anfahren
4 Mov P2          'Position 2 mittels Gelenk-Interpolation und optimaler
                  Beschleunigung/Abbremsung anfahren
5 LoadSet 1,2     'Optimale Beschleunigung/Abbremsung für Hand 1
                  (HNDDAT1) und Werkstück 2 (WRKDAT2) einstellen
6 Mov P1          'Position 1 mittels Gelenk-Interpolation und optimaler
                  Beschleunigung/Abbremsung anfahren
7 Mov P2          'Position 2 mittels Gelenk-Interpolation und optimaler
                  Beschleunigung/Abbremsung anfahren
8 Oadl Off        'Optimale Beschleunigung/Abbremsung sperren
```

Erläuterung

- Die Funktion ermöglicht die Ausführung der Roboterbewegung für verschiedene Handdaten und Werkstücke mit optimaler Beschleunigung/Abbremsung.
- Beim Programmstart wird für die Hand die maximale Last gesetzt.
- Werden mehrere Variablennamen angegeben, müssen sie durch Kommas getrennt werden.
- Stellen Sie das Gewicht, die Maße (X, Y, Z) und den Schwerpunkt (X, Y, Z) der Hand in Parameter HANDDAT (HANDDAT 1 bis 8) ein.
- Stellen Sie das Gewicht, die Maße (X, Y, Z) und den Schwerpunkt (X, Y, Z) des Werkstücks in Parameter WRKDAT (WRKDAT 1 bis 8) ein.
- Die Hand- und Werkzeugbedingungen für die optimale Beschleunigung/Abbremsung werden beim Zurücksetzen des Programms und bei Ausführung der End-Anweisung auf die Standardwerte zurückgesetzt.
- Als Standardwert wird die Handbedingung auf den Lastnennwert und die Werkzeugbedingung auf „keine“ (0 kg) gesetzt.
- Weitere Hinweise zur Einstellung der optimalen Beschleunigung/Abbremsung finden Sie in Abschn. 9.19.1 „Optimale Beschleunigung/Abbremsung“ und Kap. 9 unter dem Parameter ACCMODE.

Steht in Beziehung zu folgenden Befehlen:

Oadl, HOpen/HClose

Steht in Beziehung zu folgenden Parametern:

HNDDAT1 bis 8, WRKDAT1 bis 8, HNDHOLD1 bis 8

6.3.48 Mov (Move)

Funktion: Bewegung mit Gelenk-Interpolation

Bewegt die Handspitze mittels Gelenk-Interpolation zu einer festgelegten Position

Eingabeformat

Mov ☐ <Zielposition> [,<Abstand>][☐ TYPE ☐ <Konstante 1>,<Konstante 2>] ☐ [<Verknüpfungsbedingung>]

<Zielposition>	Legt die Zielposition als Positionskonstante, -variable oder Gelenkvariable fest
<Abstand>	Legt den Verfahrwegbetrag in Werkzeugrichtung auf der Z-Achse fest (Abstand zur Zielposition)
TYPE	
<Konstante 1>	Indirekte/direkte Anfahrt = 1/0 (Standardwert: 1)
<Konstante 2>	Ist auf „0“ festgelegt
<Verknüpfungsbedingung>	Es können die Verknüpfungen Wth und WthIf verwendet werden.

Programmbeispiel

```
1 Mov P1 TYPE 1,0                                   'Position 1 anfahren
2 Mov P2                                            'Position 2 anfahren
3 Mov (PLT 1,10),100.0 Wth M_Out(17) = 1            'Palette 1 anfahren und
                                                    Ausgangsbit 17 auf „1“
                                                    setzen
4 Mov P4 + P5,50.0 WthIf M_In(18 )= 1,M_Out(20) = 1 'Position (P4 + P5)
                                                    anfahren, wenn
                                                    Eingangsbit 18
                                                    gleich 1 ist, setze
                                                    Ausgangsbit 20 auf „1“
```

Erläuterung

- Die Gelenkwinkel jeder Achse werden am Start- und Endpunkt gleichmäßig interpoliert. Der Weg der Handspitze kann daher nicht exakt vorhergesagt werden.

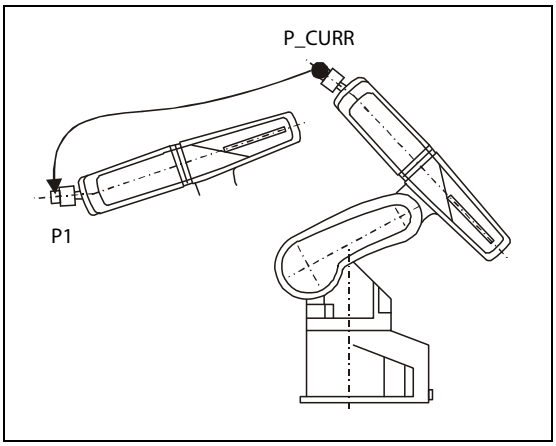


Abb. 6-13:
Verfahrweg bei Gelenk-Interpolation

R000885C

- Durch die Verknüpfungsbedingungen über die Wth- und WthIf-Anweisung können die Verfahrbewegung und die Signalausgabe synchronisiert werden.
- Über die numerische Konstante 1 zur Festlegung des Interpolationstyps wird die Interpolation der Stellung definiert.
- Bei der indirekten Anfahrt erfolgt die Anfahrt der Position exakt mit der geteachten Stellung. In Abhängigkeit der geteachten Stellung kann die direkte Anfahrt gewählt werden.
- Bei der direkten Anfahrt wird die Stellung beim Start bis zur Stellung bei Erreichen der Zielposition mit weniger Bewegungen geändert.
- Die Auswahl zwischen direkter und indirekter Anfahrt ist bei einem Bewegungsbereich der Stellungsachse von 180° oder mehr von Bedeutung.
- Liegt die Zielposition bei angewählter direkter Anfahrt außerhalb des Bewegungsbereiches, ist es möglich, dass die Achse in umgekehrter Richtung mit indirekter Anfahrt bewegt wird.
- Die numerische Konstante 2 ist für die Gelenk-Interpolation bedeutungslos.
- Dieser Befehl kann in einem kontinuierlich ausgeführten Programm nicht verwendet werden.
- Wird die Ausführung eines Mov-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden. Weiterhin erlaubt die Funktion die Fortsetzung des Verfahrwegs von der aktuellen zur Zielposition, ohne Anfahrt der Position, bei der die Verfahrbewegung unterbrochen wurde (siehe auch Abschn. 9.11).

6.3.49 Mva (Move Arch)

Funktion: Bewegung mit Bogen-Interpolation

Bewegt die Handspitze mittels Bogen-Interpolation zu einer festgelegten Position

Eingabeformat

Mva □ <Zielposition> [,<Bogennummer>]

<Zielposition>	Legt die Zielposition als Positionskonstante oder -variable oder Gelenkvariable fest
<Bogennummer>	Die Bogennummer entspricht dem mit dem Befehl Def Arch definierten Bogen 1 ≤ Bogennummer ≤ 4 Fehlt die Angabe, wird der Wert auf „1“ gesetzt.

Programmbeispiel

```
1 Def Arch 1,5,5,20,20          'Bogen definieren
2 Ovrd 100,20,20                'Übersteuerung festlegen
3 Accel 100,100,50,50,50,50     'Beschleunigung/Abbremsung festlegen
4 Mva P1,1                      'Position P1 mittels Bogen-Interpolation über den
                                 in Programmschritt 1 definierten Bogen anfahren
5 Mva P2,2                      'Position P2 mittels Bogen-Interpolation über den
                                 mit den Standardeinstellungen definierten Bogen
                                 anfahren
```

Erläuterung

Die Verfahrbewegung erfolgt von der Startposition entlang der Z-Achse an aufwärts, dann zu einer Position über der Zielposition und anschließend wieder nach unten zur Zielposition. Die Ausführung der bogenförmigen Verfahrbewegung erfolgt über eine einzelne Anweisung.

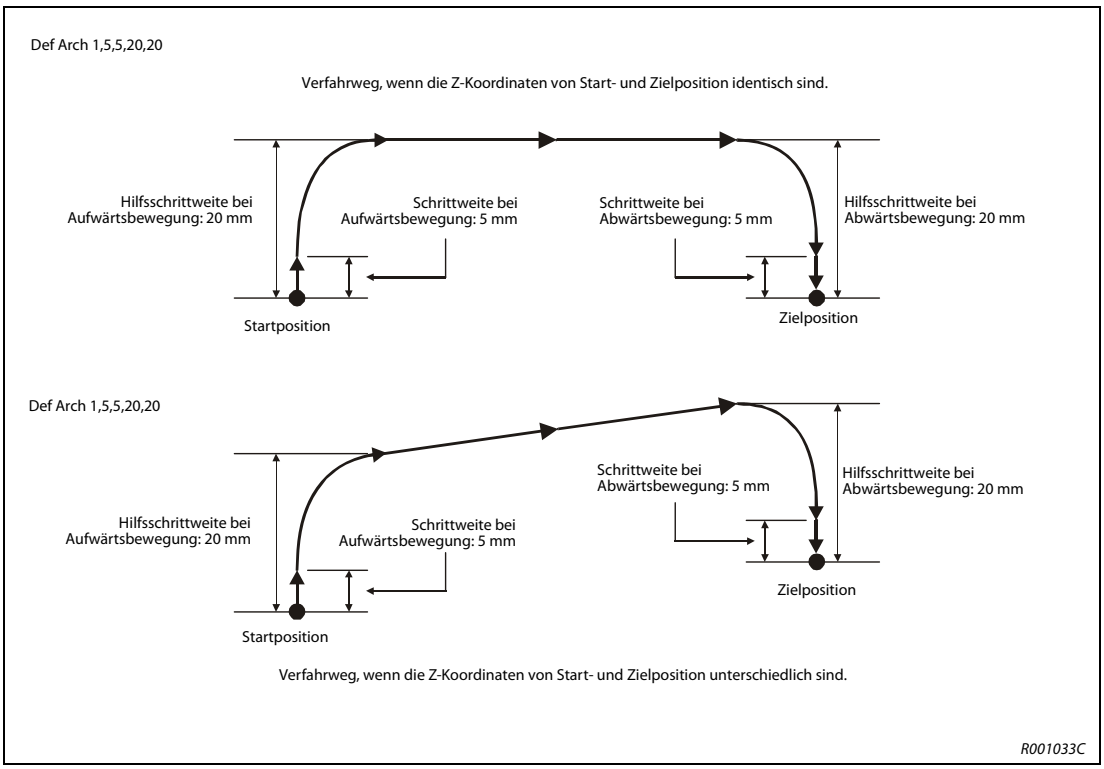

Def Arch 1,5,5,20,20
Verfahrweg, wenn die Z-Koordinaten von Start- und Zielposition identisch sind.
Hilfsschrittweite bei Aufwärtsbewegung: 20 mm
Schrittweite bei Aufwärtsbewegung: 5 mm
Schrittweite bei Abwärtsbewegung: 5 mm
Hilfsschrittweite bei Abwärtsbewegung: 20 mm
Startposition
Zielposition
Def Arch 1,5,5,20,20
Schrittweite bei Abwärtsbewegung: 5 mm
Hilfsschrittweite bei Abwärtsbewegung: 20 mm
Hilfsschrittweite bei Aufwärtsbewegung: 20 mm
Schrittweite bei Aufwärtsbewegung: 5 mm
Zielposition
Startposition
Verfahrweg, wenn die Z-Koordinaten von Start- und Zielposition unterschiedlich sind.
R001033C

Abb. 6-14: *Beispiele zur Bogen-Interpolation*

- Wird der Befehl Mva ohne vorherige Definition des Bogens über den Def-Arch-Befehl ausgeführt, erfolgt die Verfahrbewegung entsprechend dem über die Parameter festgelegten Bogen (siehe auch Abschn. 6.3.20 „Def Arch").
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.
- Wird die Ausführung eines Mva-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Eine Änderung dieser Funktion sowie die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).

ACHTUNG:

Der genaue Verfahrweg des Roboters kann sich in Abhängigkeit der Geschwindigkeit ändern. Besonders im Bereich der Eckpunkte kann der Verfahrweg variieren. Verfahren Sie den Roboter beim Start des Automatikbetriebs daher erst mit niedriger Geschwindigkeit. Steigern Sie die Geschwindigkeit allmählich und achten Sie darauf, dass keine Kollisionen mit umliegenden Einheiten auftreten.

Steht in Beziehung zu folgenden Befehlen:

Def Arch, Accel, Ovrd

6.3.50 Mvc (Move C)

Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch Startposition, Zwischenposition 1, Zwischenposition 2 und Startposition festgelegten Kreises

Eingabeformat

Mvc ☐ <Startposition>,<Zwischenposition 1>,<Zwischenposition 2>
☐ [<Verknüpfungsbedingung>]

<Startposition>	Legt den Start- und Endpunkt des Kreises fest
<Zwischenposition 1>	Legt die erste Zwischenposition auf dem Kreisumfang fest
<Zwischenposition 2>	Legt die zweite Zwischenposition auf dem Kreisumfang fest
<Verknüpfungsbedingung>	Es können die Verknüpfungen Wth und WthIf verwendet werden.

Programmbeispiel

```
1 Mvc P1,P2,P3                                   'Bewegung entlang des durch P1, P2 und
                                                 P3 festgelegten Kreises
2 Mvc P1,J2,P3                                   'Bewegung entlang des durch P1, J2 und
                                                 P3 festgelegten Kreises
3 Mvc P1,P2,P3 Wth M_Out(17) = 1                 'Bewegung entlang des durch P1, P2 und
                                                 P3 festgelegten Kreises und Setzen des
                                                 Ausgangsbits 17 auf „1“
4 Mvc P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1   'Bewegung entlang des
                                                 durch P3, (PLT 1,5) und
                                                 P4 festgelegten Kreises
                                                 und Setzen des
                                                 Ausgangsbits 21, falls
                                                 Eingangsbit 20 gleich 1
```

Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Handspitze des Roboters auf dem Kreisumfang des durch die 3 Punkte festgelegten Kreises (360°).

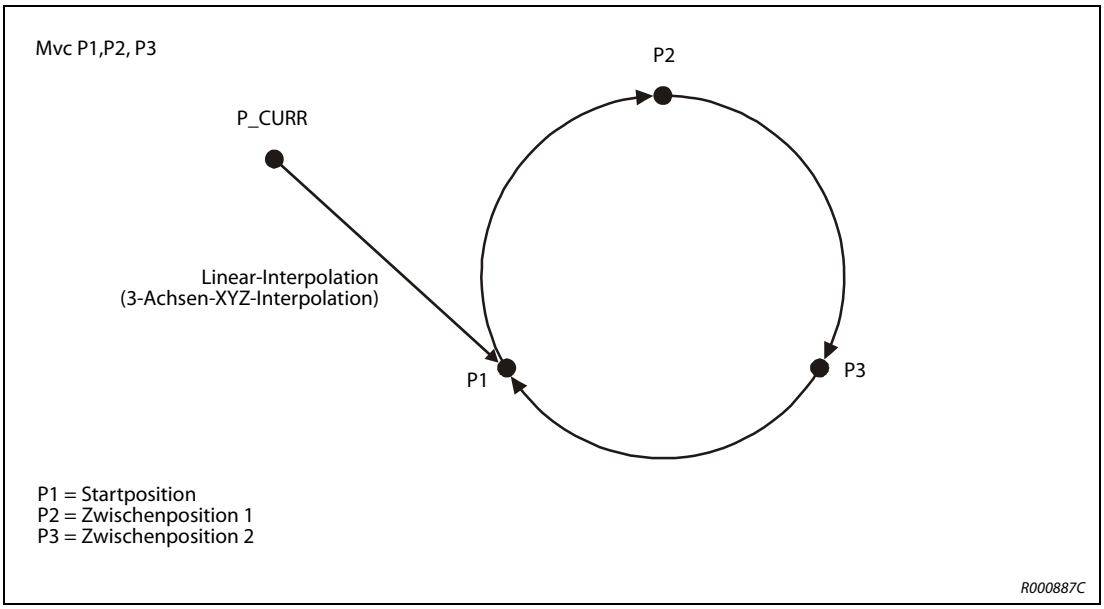


Abb. 6-15: *Beispiel zur Kreis-Interpolation über zwei Zwischenpositionen*

- Während der Kreis-Interpolation bleibt die Orientierung des Roboters im Startpunkt unverändert. Die Orientierungen bei Zwischenposition 1 und Zwischenposition 2 sind nicht definiert.
- Entspricht die momentane Position nicht der Startposition, fährt der Roboter die Startposition mittels Linear-Interpolation (3-Achsen-XYZ-Interpolation) an und führt anschließend die Kreis-Interpolation aus.
- Wird die Ausführung eines Mvc-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Der Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

6.3.51 Mvr (Move R)

Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch Startposition, Zwischenposition und Endposition festgelegten Kreisbogens.

Eingabeformat

Mvr ☐ <Startposition>,<Zwischenposition>,<Endposition>
[☐ TYPE ☐ <Konstante 1>,<Konstante 2>] ☐ [<Verknüpfungsbedingung>]

<Startposition>	Legt den Startpunkt des Kreises fest
<Zwischenposition>	Legt die Zwischenposition auf dem Kreisumfang fest
<Endposition>	Legt die Endposition auf dem Kreisumfang fest
TYPE <Konstante 1> <Konstante 2>	Ermöglicht das Durchfahren des singulären Punktes 1/0 = direkte/indirekte Anfahrt (Standardwert: 0) Drehung/3-Achsen-XYZ/Durchfahren des singulären Punktes = 0/1/2 (Standardwert: 0)
<Verknüpfungsbedingung>	Es können die Verknüpfungen Wth und WthIf verwendet werden.

Programmbeispiel

```
1 Mvr P1,P2,P3                                   'Bewegung entlang des durch P1, P2 und
                                                  P3 festgelegten Kreisbogens
2 Mvr P1,J2,P3                                   'Bewegung entlang des durch P1, J2 und
                                                  P3 festgelegten Kreisbogens
3 Mvr P1,P2,P3 Wth M_Out(17) = 1                 'Bewegung entlang des durch P1, P2 und
                                                  P3 festgelegten Kreisbogens und Setzen des
                                                  Ausgangsbits 17 auf „1“
4 Mvr P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1
                                                 'Bewegung entlang des
                                                  durch P3, (PLT 1,5) und
                                                  P4 festgelegten Kreisbogens
                                                  und Setzen des
                                                  Ausgangsbits 21, falls
                                                  Eingangsbit 20 gleich 1
```

Erläuterung

- Mittels Kreis-Interpolation bewegt sich der Roboterarm auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist.

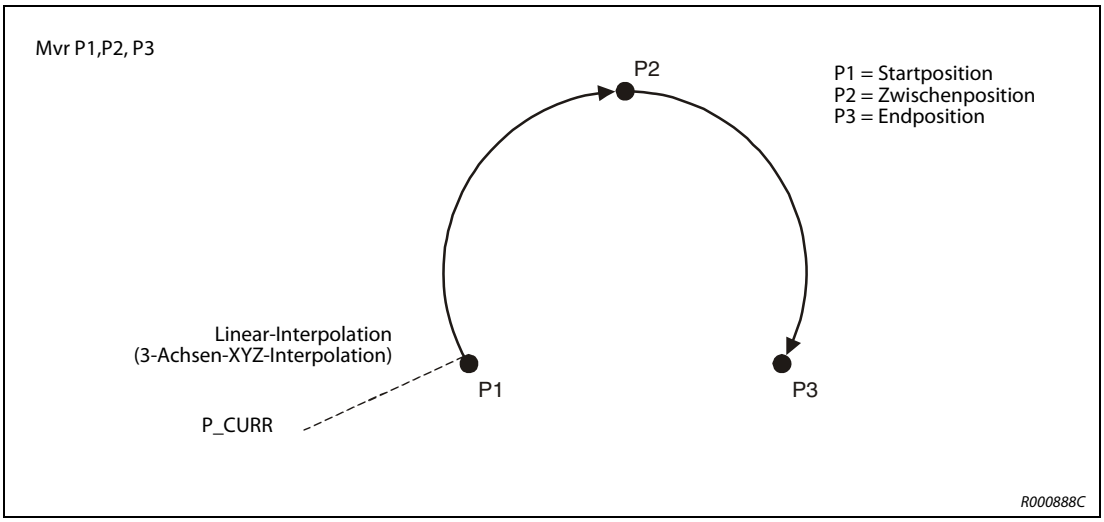


Abb. 6-16: *Beispiel zur Kreis-Interpolation über eine Zwischenposition*

- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung der Zwischenposition hat keinen Einfluss.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter die Startposition mittels Linear-Interpolation an.
- Wird die Ausführung eines Mvr-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Weichen die Stellungsmerker der Start- und Endposition für die Interpolationsmethode „Drehung“ (Konstante 2 = 0) voneinander ab, erfolgt eine Fehlermeldung.
- Der Roboter verfährt mit Linear-Interpolation, wenn zwei der drei Positionen gleich sind oder alle Positionen auf einer Geraden liegen. Es erfolgt keine Fehlermeldung.
- Wird über die Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.
- Dieser Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

6.3.52 Mvr2 (Move R2)

Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition
Der Kreisbogen wird durch die Startposition, die Referenzposition und die Endposition festgelegt. Die Roboterbewegung geht dabei nicht durch den Referenzpunkt.

Eingabeformat

Mvr2 ☐ <Startposition>,<Endposition>,<Referenzposition>
[☐ TYPE ☐ <Konstante 1>,<Konstante 2>] ☐ [<Verknüpfungsbedingung>]

<Startposition>	Legt den Startpunkt des Kreises fest
<Endposition>	Legt die Endposition auf dem Kreisumfang fest
<Endposition>	Legt die Referenzposition auf dem Kreisumfang fest
TYPE <Konstante 1> <Konstante 2>	Ermöglicht das Durchfahren des singulären Punktes 1/0 = direkte/indirekte Anfahrt (Standardwert: 0) Drehung/3-Achsen-XYZ/Durchfahren des singulären Punktes = 0/1/2 (Standardwert: 0)
<Verknüpfungsbedingung>	Es können die Verknüpfungen Wth und WthIf verwendet werden.

Programmbeispiel

```
1 Mvr2 P1,P2,P3                                          'Bewegung entlang des durch P1, P2 und
                                                         P3 festgelegten Kreisbogens
2 Mvr2 P1,J2,P3                                          'Bewegung entlang des durch P1, J2 und
                                                         P3 festgelegten Kreisbogens
3 Mvr2 P1,P2,P3 Wth M_Out(17) = 1                        'Bewegung entlang des durch P1, P2 und
                                                         P3 festgelegten Kreisbogens und Setzen des
                                                         Ausgangsbits 17 auf „1“
4 Mvr2 P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1  'Bewegung entlang des
                                                         durch P3, (PLT 1,5) und
                                                         P4 festgelegten Kreisbogens
                                                         und Setzen des
                                                         Ausgangsbits 21, falls
                                                         Eingangsbit 20 gleich 1
```

Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Roboterhand auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist. Die Bewegung geht nicht durch die Referenzposition.

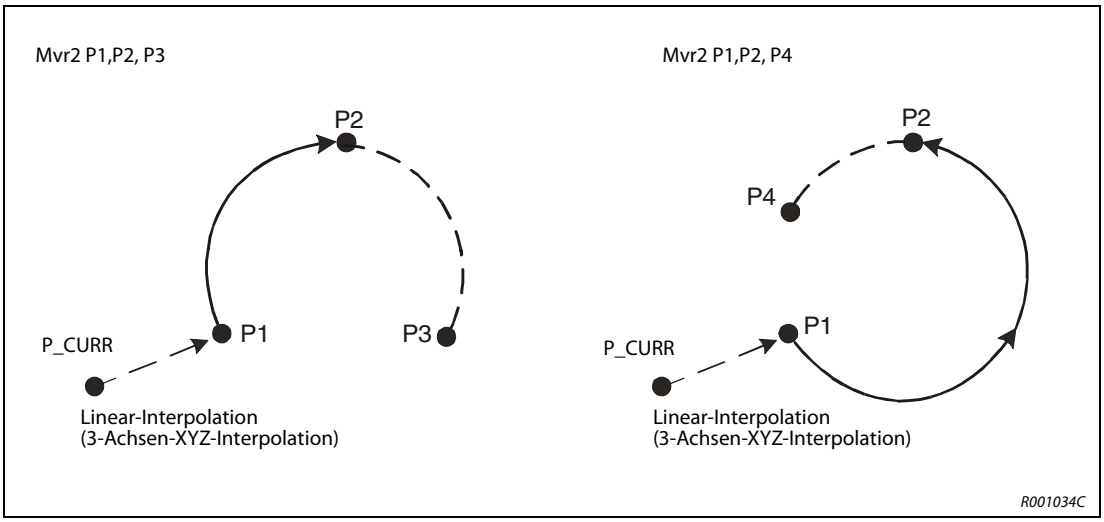


Abb. 6-17: *Beispiel zur Kreis-Interpolation über einen Referenzpunkt*

- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung des Referenzpunktes hat keinen Einfluss.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter automatisch die Startposition mittels Linear-Interpolation an.
- Wird die Ausführung eines Mvr2-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Der Roboter bewegt sich in die Richtung entlang des Kreisbogens, die nicht durch die Referenzposition geht.
- Weichen die Stellungsmerker der Start- und Endposition für die Interpolationsmethode „Drehung“ (Konstante 2 = 0) voneinander ab, erfolgt eine Fehlermeldung.
- Der Roboter verfährt mit Linear-Interpolation, wenn zwei der drei Positionen gleich sind oder alle Positionen auf einer Geraden liegen. Es erfolgt keine Fehlermeldung.
- Wird über die Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.
- Der Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

6.3.53 Mvr3 (Move R3)

Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition. Der Kreisbogen wird durch die Startposition, den Mittelpunkt und die Endposition festgelegt.

Eingabeformat

```
Mvr3 ☐ <Startposition>,<Endposition>,<Mittelpunkt>
        [☐ TYPE ☐ <Konstante 1>,<Konstante 2>] ☐ [<Verknüpfungsbedingung>]
```

<Startposition>	Legt den Startpunkt des Kreises fest
<Endposition>	Legt die Endposition auf dem Kreisumfang fest
<Mittelpunkt>	Legt den Mittelpunkt des Kreises fest
TYPE <Konstante 1> <Konstante 2>	Ermöglicht das Durchfahren des singulären Punktes 1/0 = direkte/indirekte Anfahrt (Standardwert: 0) Drehung/3-Achsen-XYZ/Durchfahren des singulären Punktes = 0/1/2 (Standardwert: 0)
<Verknüpfungsbedingung>	Es können die Verknüpfungen Wth und WthIf verwendet werden.

Programmbeispiel

```
1 Mvr3 P1,P2,P3                                   'Bewegung entlang des durch P1, P2 und
                                                  P3 festgelegten Kreisbogens
2 Mvr3 P1,J2,P3                                   'Bewegung entlang des durch P1, J2 und
                                                  P3 festgelegten Kreisbogens
3 Mvr3 P1,P2,P3 Wth M_Out(17) = 1                 'Bewegung entlang des durch P1, P2 und
                                                  P3 festgelegten Kreisbogens und Setzen des
                                                  Ausgangsbits 17 auf „1“
4 Mvr3 P3,(Plt 1,5),P4 WthIf M_In(20) = 1,M_Out(21) = 1
                                                  'Bewegung entlang des
                                                  durch P3, (PLT 1,5) und
                                                  P4 festgelegten Kreisbogens
                                                  und Setzen des
                                                  Ausgangsbits 21, falls
                                                  Eingangsbit 20 gleich 1
```

Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Roboterhand auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist.

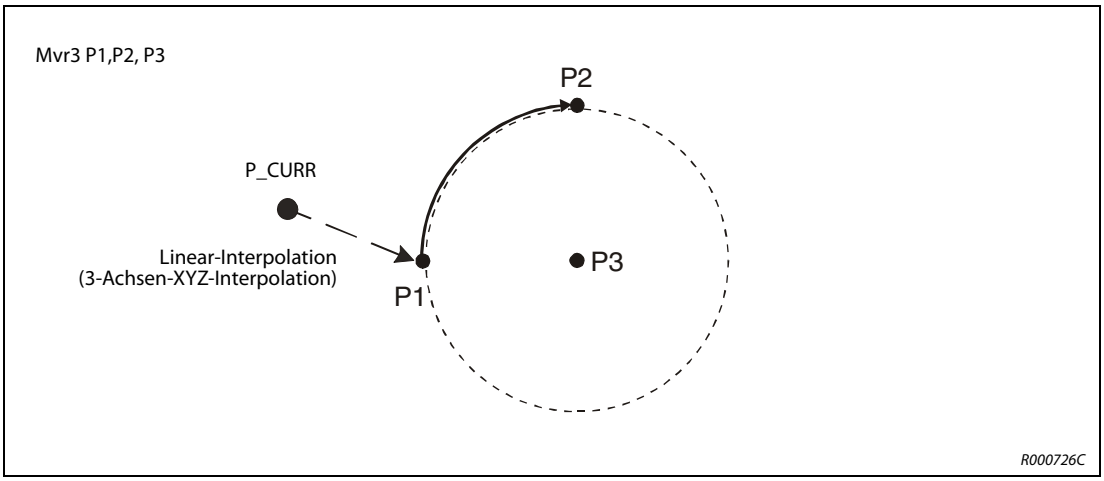


Abb. 6-18: *Beispiel zur Kreis-Interpolation über einen Mittelpunkt*

- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung des Referenzpunktes hat keinen Einfluss.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter automatisch die Startposition mittels Linear-Interpolation an.
- Wird die Ausführung eines Mvr3-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Weichen die Stellungsmerker der Start- und Endposition für die Interpolationsmethode „Drehung“ (Konstante 2 = 0) voneinander ab, erfolgt eine Fehlermeldung.
- Wird über die Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.
- Der Zentriwinkel vom Start- bis zum Endpunkt liegt zwischen 0 und 180°.
- Legen Sie die Positionen so fest, dass die Differenz zwischen Mittelpunkt und Endpunkt und die Differenz zwischen Mittelpunkt und Startpunkt größer als 0,01 mm sind.
- Fallen der Start- und der Mittelpunkt oder der End- und der Mittelpunkt zusammen, oder liegen alle Positionen auf einer Geraden, erfolgt eine Fehlermeldung.
- Stimmen der Start- und Endpunkt oder alle 3 Punkte überein, erfolgt keine Fehlermeldung. Der nächste Befehl wird ausgeführt. Ändert sich in dieser Zeit die Stellung, wird nur die Stellung interpoliert.
- Der Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

6.3.54 Mvs (Move S)

Funktion: geradlinige Bewegung

Bewegt die Handspitze mittels Linear-Interpolation zur festgesetzten Position

Eingabeformat 1

```
Mvs ☐ <Zielposition> [,<Abstand>]
    [☐ TYPE ☐ <Konstante 1>,<Konstante 2>] ☐ [<Verknüpfungsbedingung>]
```

Eingabeformat 2

```
Mvs ☐ ,<Verfahrbetrag>
    [☐ TYPE ☐ <Konstante 1>,<Konstante 2>] ☐ [<Verknüpfungsbedingung>]
```

<Zielposition>	Legt die Zielposition fest
<Abstand>	Legt den Verfahrwegbetrag in Werkzeugrichtung auf der Z-Achse fest (Abstand zur Zielposition/±-Richtung)
TYPE <Konstante 1> <Konstante 2>	Ermöglicht das Durchfahren des singulären Punktes 1/0 = direkte/indirekte Anfahrt (Standardwert: 0) Drehung/3-Achsen-XYZ/Durchfahren des singulären Punktes = 0/1/2 (Standardwert: 0)
<Verknüpfungsbedingung>	Es können die Verknüpfungen Wth und WthIf verwendet werden.
<Verfahrbetrag>	Legt den Verfahrwegbetrag von der Augenblicksposition in Werkzeuglängsrichtung auf der Z-Achse fest (Abstand zur Zielposition) Bei einem positiven Betrag bewegt sich die Handspitze in Werkzeuglängsrichtung nach vorne. Bei Angabe eines negativen Verfahrwegbetrags wird die Handspitze in Werkzeuglängsrichtung zurückgefahren.

Programmbeispiel

```
1 Mvs P1                                          'Position P1 mittels
                                                  Linear-Interpolation anfahren
2 Mvs P1,–100.0 Wth M_Out(17) = 1                 'Position anfahren, die um
                                                  100 mm in Werkzeuglängsrichtung
                                                  entfernt von P1 liegt und Ausgangsbit 17
                                                  auf „1“ setzen
3 Mvs P4+P5,–50.0 WthIf M_In(18) = 1,M_Out(20) = 1
                                                  'Position anfahren, die um
                                                  50 mm in Werkzeuglängsrichtung
                                                  entfernt von (P4+P5) liegt und
                                                  Ausgangsbit 20 auf „1“ setzen,
                                                  falls Eingangsbit 18 gleich 1 ist
4 Mvs ,–50                                        'Position anfahren, die 50 mm in
                                                  Werkzeuglängsrichtung von der aktuellen
                                                  Position entfernt ist
```

Erläuterung

- Dieser Befehl verfährt die Handspitze entlang einer geraden Linie zur festgelegten Position.
- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert.
- Erfolgt die Verfahrbewegung durch Angabe des Abstands oder Verfahrwegbetrags im Werkzeugkoordinatensystem hängt die Richtung des Verfahrwegs vom Werkzeugkoordinatensystem des Roboters ab (siehe auch Abschn. 9.7).
  Folgende Abbildung zeigt den Verfahrweg des Roboters RV-6SD bei Linear-Interpolation.

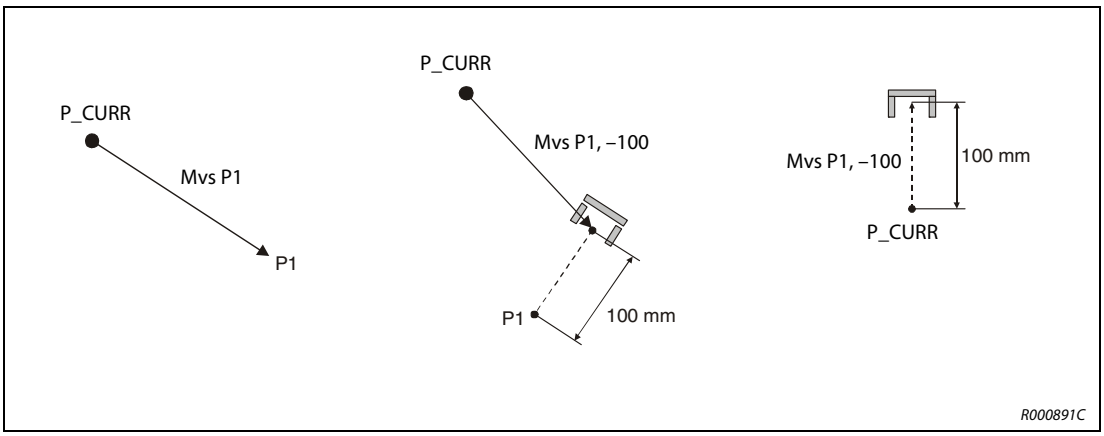


Abb. 6-19: *Beispiel zur Linear-Interpolation*

- Wird die Ausführung eines Mvs-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Eine Änderung dieser Funktion sowie die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.
- Weichen die Stellungsmerker der Start- und Endposition für die Interpolationsmethode „Drehung“ (Konstante 2 = 0) voneinander ab, erfolgt eine Fehlermeldung.
- Wird über die Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.

Singulärer Punkt

Singuläre Punkte markieren im Allgemeinen physikalische Umschlagpunkte oder kritische technische Daten, wie z. B. das Reißen eines auf Zug beanspruchten Seils.

Die Bedeutung singulärer Punkte beim Roboter wird nachfolgend anhand eines 6-achsigen Roboterarms erläutert.

Eine Bewegung von Stellung A über Stellung B zur Stellung C kann nicht mittels Linear-Interpolation (Mvs) erfolgen. Diese Einschränkung gilt nur, wenn der Winkel der J4-Achse in allen drei Stellungen (A, B und C) 0° beträgt.

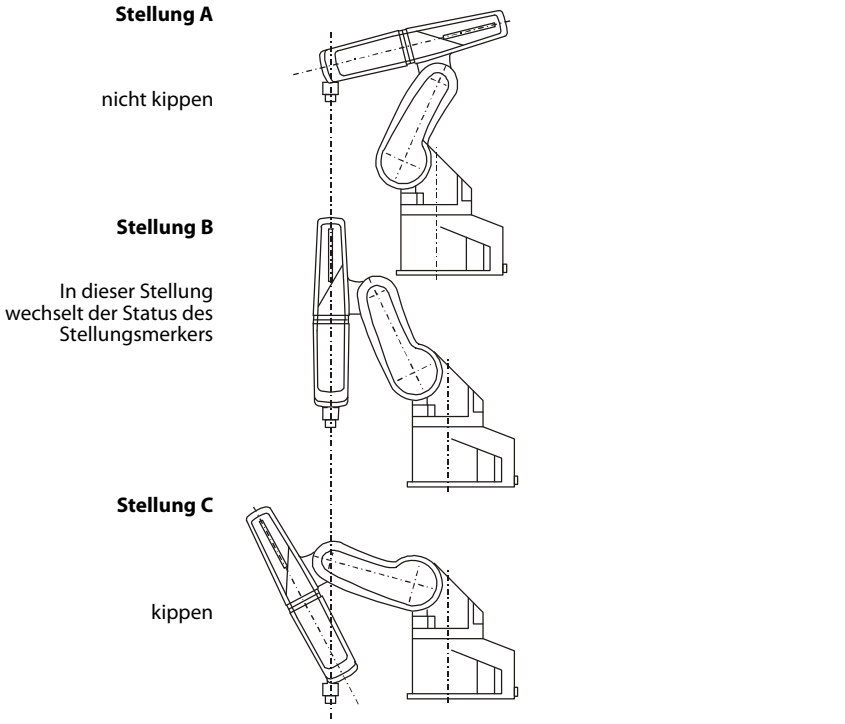


Abb. 6-20: *Singulärer Punkt*

Der Stellungsmerkerstatus der J5-Achse (Handgelenkneigungsachse) in Stellung A ist „nicht kippen" und „kippen" in Stellung C. Weiterhin ist in Stellung B das Handgelenk gestreckt und die Achsen J4 und J6 liegen auf einer Linie. Unter diesen Bedingungen ist keine Berechnung einer Position mittels Linear-Interpolation möglich.

Bei diesen Stellungen ist eine 3-Achsen-XYZ-Linear-Interpolation möglich, wenn im Mvs-Befehl die Festlegung „TYPE 0, 1" erfolgt. Da jedoch die Drehwinkel der Achsen J4, J5 und J6 am Start- und am Endpunkt gleichmäßig interpoliert werden, bleibt die Stellung streng genommen nicht erhalten. Die Roboterhand wird sich beim Verfahren von Stellung A nach C vor und zurück bewegen. Zu Minimierung der Handstellungsänderung kann in der Mitte des Verfahrwegs eine Position hinzugefügt werden.

Ein weiterer singulärer Punkt ist erreicht, wenn sich der Mittelpunkt der J5-Achse mit nach oben gerichtetem Handflansch über dem Nullpunkt befindet. In diesem Fall liegen die Achsen J1 und J6 auf einer Geraden und eine Berechnung der Roboterposition ist nicht möglich.

6.3.55 MvTune (Move Tune)

Funktion: Bewegungsmodus

Legt einen von vier möglichen Bewegungsmodi fest
Durch die Auswahl des passenden Modus lässt sich der Roboter optimal an die Anwendung anpassen. Die Betriebseigenschaften des Roboters werden dabei unter Berücksichtigung der mit dem LoadSet-Befehl eingestellten Hand- und Werkstückbedingungen optimiert.
Stellen Sie das Gewicht, die Abmessungen und den Schwerpunkt für die verwendete Hand korrekt ein.

Eingabeformat

MvTune □ <Bewegungsmodus>

<Bewegungsmodus> Legt den Bewegungsmodus über eine Konstante oder numerische Variable fest
1 ≤ Bewegungsmodus ≤ 4
1: Standardmodus (Standardwert)
2: Hochgeschwindigkeitsmodus
3: Bahnverfolgungsmodus
4: Modus zur Vibrationsunterdrückung

Bewegungsmodus		Beschreibung
1	Standardmodus (Standardwert)	Ab Werk ist der Standardmodus voreingestellt. Der Standardmodus ist auf alle herkömmlichen Anwendungen abgestimmt.
2	Hochgeschwindigkeitsmodus	Der Hochgeschwindigkeitsmodus ist so ausgelegt, dass die Zielposition möglichst schnell erreicht wird. Verwenden Sie den Modus, wenn die Positionierzeit verkürzt und der Durchsatz erhöht werden soll. (Anwendungen: Nachführbetrieb, Palettierung, Entnahmen usw.)
3	Bahnverfolgungsmodus	In diesem Modus wird die Genauigkeit einer interpolierten Bewegung bei der Bahnverfolgung erhöht. Verwenden sie diesen Modus, wenn es auf eine hohe Genauigkeit bei der Bahnverfolgung ankommt. (Anwendungen: Kleben, Schweißen, Entgraten usw.)
4	Modus zur Vibrationsunterdrückung ①	In diesem Modus werden Vibrationen des Roboterarms (Resonanzen) unterdrückt. Verwenden Sie diesen Modus in Applikationen, in denen beim Transport von Werkstücken Vibrationen auftreten können. (Anwendungen: Wafertransportsystem, Transport von Präzisionsteilen usw.)

Tab. 6-9: *Bewegungsmodi des Befehls MvTune*

① Der Modus zur Vibrationsunterdrückung kann bei den Geräten CRnD-700 ab Software-Version P7 und bei den Geräten CRnQ-700 ab Software-Version N7 verwendet werden.

Programmbeispiel

```
1 LoadSet 1,1      'Optimale Beschleunigung/Abbremsung für Hand 1
                   (HNDDAT1) und Werkstück 1 (WRKDAT1) einstellen
2 MvTune 2         'Hochgeschwindigkeitsmodus festlegen
3 Mov P1           'Position 1 mittels Gelenk-Interpolation im
                   Hochgeschwindigkeitsmodus anfahren
4 Mvs P2           'Position 2 mittels Linear-Interpolation im
                   Hochgeschwindigkeitsmodus anfahren
5 MvTune 3         'Bahnverfolgungsmodus festlegen
6 Mvs P3           'Position 3 mittels Linear-Interpolation im
                   Bahnverfolgungsmodus anfahren
```

Erläuterung

- Durch diesen Befehl werden die Betriebseigenschaften des Roboters unter Berücksichtigung der mit dem LoadSet-Befehl eingestellten Hand- und Werkstückbedingungen optimiert. Sind die Hand- und Werkstückbedingungen nicht korrekt eingestellt, erzielt der Roboter eventuell keine zufriedenstellenden Ergebnisse. (Weitere Hinweise finden Sie auch in Abschn. 9.19 „Hand- und Werkstückbedingung“.)
- Direkt nach dem Einschalten ist der Standardmodus aktiv.
- Bei Beendigung eines Programms (End-Befehl, Reset nach einem Interrupt) kehren die Betriebseigenschaften in den Standardmodus zurück. Nach dem End-Befehl in einem durch den CallP-Befehl aufgerufenen Unterprogramm bleibt der aktuelle Betriebsmodus erhalten.
- Der Bewegungsmodi unterscheiden sich durch folgende Eigenschaften vom Standardmodus.

Bewegungsmodus	Vergleichskriterien			
	Zeit bis zum Erreichen der Zielposition	Verfahrwegstreue ①	Vibrationsunterdrückung ②	Lastfaktor ③
Standardmodus	○	○	○	○
Hochgeschwindigkeitsmodus	◎	○ +	○	△
Bahnverfolgungsmodus	△	◎	○	○ +
Modus für Vibrationsunterdrückung	○ –	○ –	◎	○

Tab. 6-10: *Vergleich der Bewegungsmodi*

◎ : besser
○ + : etwas besser
○ : gleich
○ – : etwas schlechter
△ : schlechter

① Fähigkeit des Roboters dem Befehlswert zu folgen
② Fähigkeit des Roboters, Störungen zu unterdrücken, die Vibrationen verursachen
③ Vergleich der Wärmeentwicklung des Motors

- Ist die optimale Beschleunigung/Abbremsung (Oadl-Befehl und Parameter ACCMOD) deaktiviert, so wird sie durch die Verwendung des MvTune-Befehls aktiviert. Durch die Ausführung des Befehls Oadl Off nach dem Befehl Mv Tune wird die optimale Beschleunigung/Abbremsung deaktiviert, der aktuelle Bewegungsmodus jedoch bleibt erhalten.
- Verglichen mit dem Standardmodus können im Hochgeschwindigkeitsmodus eher Vibrationen auftreten. Sind die Vibrationen nicht tolerabel, wählen Sie den Standardmodus.
- Der Bahnverfolgungsmodus erzielt im Bereich kleiner und mittlerer Geschwindigkeiten optimale Ergebnisse. Deshalb können bei Verfahrbewegungen, die engen Kreisbahnen folgen, höhere Vibrationen als im Standardmodus auftreten. Stören die Vibrationen, verringern Sie die Geschwindigkeit mit dem Spd-Befehl.
- In Abhängigkeit der Betriebsbedingungen kann sich im Modus für Vibrationsunterdückung die Zeit bis zum Erreichen der Zielposition vergrößern. Bei Verfahrbewegungen, die mit einer Bedingung abgeschlossen werden (z. B. Fine-Befehl) kann sich dadurch die Zykluszeit vergrößern. Je größer die Masse des Werkstücks, desto größer der Zeitzuwachs.
- Der Befehl kann nicht im JOG-Betrieb verwendet werden.

Steht in Beziehung zu folgenden Befehlen:

LoadSet, Oadl, Prec

Steht in Beziehung zu folgenden Parametern:

ACCMODE, HNDDAT0 bis 8, WRKDAT0 bis 8

6.3.56 Mxt (Move External)

Funktion: Externe Steuerung

Ermöglicht eine Steuerung in Echtzeit über eine Ethernet-Schnittstelle
Mit jedem Taktzyklus des Steuergeräts (ca. 7.1 ms) können Positionsdaten von einer externen Quelle abgefragt werden, die die Roboterbewegung direkt steuern.

Eingabeformat

Mxt <Dateinummer>,<Positionsdatentyp der Antwordaten>[,<Zeitkonstante des Filters>]

<Dateinummer>	Gibt die Nummer einer mit dem Open-Befehl geöffneten Datei an 1 ≤ Dateinummer ≤ 8 Wurde keine Datei mit dem Open-Befehl geöffnet, erfolgt eine Fehlermeldung und es ist keine Kommunikation möglich. Daten, die von einer anderen als der angegeben Quelle empfangen werden, werden ignoriert.
<Positionsdatentyp der Antwortdaten>	Legt den Positionsdatentyp fest, der vom Personal-computer empfangen werden kann Es können Daten des Typs XYZ/Gelenk/Motorimpuls festgelegt werden. 0: XYZ-Koordinatendaten 1: Gelenkkoordinatendaten 2: Motorimpulskoordinatendaten
<Zeitkonstante des Filters>	Bei einer Einstellung von „0“ ist das Filter nicht aktiviert. Erfolgt keine Angabe, wird der Wert „0“ gesetzt. Die empfangenen Positionsdaten werden gefiltert und steuern die Servomotoren.

Programmbeispiel

```
1 Open "ENET:192.168.0.2" As #1   'Öffnet den Ethernet-Kommunikationskanal als Datei Nr. 1
2 Mov P1                          'Position 1 mittels Gelenk-Interpolation anfahren
3 Mxt1,1,50                       'Bewegt den Roboter über externe Signale mit einer
                                  Filterzeitkonstanten von 50 ms
4 Mov P1                          'Position 1 mittels Gelenk-Interpolation anfahren
5 Hlt                             'Stoppt den Roboter
```

Erläuterung

- Nach Ausführung des Befehls Mxt können die Daten zur Steuerung der Roboterbewegung von einem am Netzwerk angeschlossenen Personalcomputer empfangen werden (One-to-one-Verbindung).
- Innerhalb eines Taktzyklus des Steuergeräts (ca. 7,1 ms) kann ein Bewegungsbefehl empfangen und verarbeitet werden.
- Nach Ausführung des Befehls wechselt das Steuergerät in den Bereitschaftsstatus zum Empfang von Befehlsdaten. Der Handzustand zum Aufnehmen oder Ablegen von Werkstücken ist in den Parametern HNDHOLD1 bis 8 festgelegt.
- Hat das Steuergerät die Daten vom Personalcomputer empfangen, gibt es die Befehlswerte während des nächsten Verarbeitungszyklus an die Servomotoren weiter.
- Nachdem die Befehlswerte an die Servomotoren gesendet wurden, werden die Statusinformationen des Steuergerätes, wie die aktuelle Position, zum Personalcomputer übertragen.
- Das Steuergerät schickt nur dann Antwortdaten an den Personalcomputer, wenn die Befehlswerte vom Personalcomputer an das Steuergerät übertragen werden.
- Erfolgt keine Übertragung der Daten, verbleibt der Roboter in der aktuellen Position.
- Hat das Steuergerät vom Personalcomputer den Echtzeit-Steuerungs-Ende-Befehl empfangen, wird der Mxt-Befehl beendet.
- Bei einer Unterbrechung des Betriebs über das Bedienfeld des Steuergerätes oder ein externes Signal, wird auch der Befehl Mxt unterbrochen und der Empfang bzw. das Senden der Daten wird solange eingestellt, bis ein Neustart erfolgt.
- Die Wartezeit kann über den Parameter MXTTOUT eingestellt werden.
- Ein beliebig festgelegtes (Kopf-Bit, Bitbreite) Ein-/Ausgangssignal kann parallel mit den Positionsdaten übertragen und empfangen werden.
- Um die Übertragung eines Bewegungsbefehls innerhalb eines Taktzyklus zu ermöglichen, muss ein Personalcomputer mit ausreichend hoher Geschwindigkeit verwendet werden.
- Weitere Hinweise finden Sie im Abschn. 9.17 „Kommunikationseinstellungen (Ethernet)“.
  Als Personalcomputer empfehlen wir einen Rechner mit Windows ® NT oder 2000 mit mindestens 450 Mhz.

Steht in Beziehung zu folgenden Befehlen:

Open

6.3.57 Oadl (Optimum Acceleration/Deceleration)

Funktion: Optimale Beschleunigung/Abbremsung

Legt die optimale Beschleunigungs-/Abbremszeit in Abhängigkeit von der Lasteinstellung der Hand fest
Dadurch ist eine Verkürzung der Taktzeiten möglich.

Ist die optimale Beschleunigung/Abbremsung freigegeben kann die Beschleunigungs-/Bremszeit wie folgt berechnet werden:

Beschleunigungs-/ Bremszeit [s]	=	Optimale Beschleunigungs-/ Bremszeit [s]	×	Einstellwert des Accel-Befehls [%]	×	M_SetAdl [%]

Eingabeformat

Oadl □ <On/Off>

<On/Off>	On: Einstellung für die optimale Beschleunigung/Abbremsung freigegeben Off: Einstellung für die optimale Beschleunigung/Abbremsung sperren

Programmbeispiel

```
1 LoadSet 1,1     'Optimale Beschleunigung/Abbremsung für Hand 1
                  und Werkstück 1 einstellen
2 Oadl On         'Optimale Beschleunigung/Abbremsung freigeben
3 Mov P1          'Position 1 anfahren
4 Mov P2          'Position 2 anfahren
5 HOpen 1         'Öffnet Hand 1
6 Mov P3          'Position 3 anfahren
7 HClose 1        'Schließt Hand 1
8 Mov P4          'Position 4 anfahren
9 Oadl Off        'Sperrt die optimale Beschleunigung/Abbremsung
```

Parameter HNDHOLD ist auf „0, 1“ gesetzt.

Erläuterung

- Der Roboter bewegt sich mit der optimalen Beschleunigung/Abbremsung für die über den LoadSet-Befehl eingestellten Bedingungen für die Hand und das Werkstück.

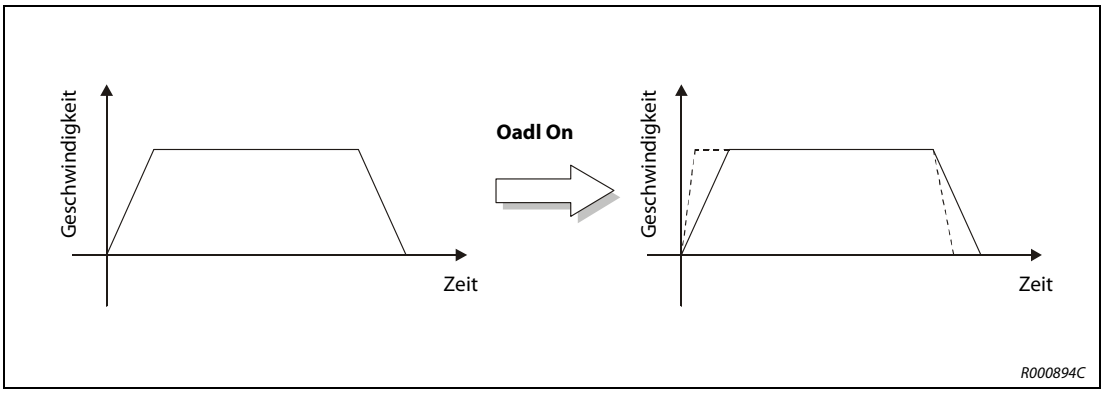


Abb. 6-21: *Optimale Beschleunigung/Abbremsung*

- Die Zuordnungen für das Öffnen/Schließen einer Hand und den Befehlen HOpen oder HClose erfolgen über die Parameter HNDHOLD 1 bis 8.
- Die Oadl-Standardeinstellung kann über Parameter ACCMODE verändert werden (siehe auch Tab. 9-3 „Signalparameter").
- Der Oadl-Befehl ist so lange aktiv (Oadl On), bis er auf Off gesetzt wird (Oadl Off) oder die End-Anweisung ausgeführt wird.
- In Abhängigkeit der Hand- und Werkstückbedingungen kann sich die Verfahrbewegung auch verlangsamen.
- Der Betrieb mit optimaler Beschleunigung/Abbremsung erfolgt über die Befehle LoadSet und Oadl sowie die Parameter HNDDAT1(0) bis 8 und WRKDAT1(0) bis 8 (siehe auch Abschn. 9.19.1).
- Die Einstellzeit für die optimale Beschleunigungs-/Abbremszeit jeder Achse ist im Parameter JADL fesgelegt. Der Parameter ist nur für die Robotermodelle der S- und SD-Serie verfügbar und vom jeweiligen Roboter abhängig (siehe auch Abschn. 9.2).

Steht in Beziehung zu folgenden Befehlen:

Accel, LoadSet, HOpen/HClose

Steht in Beziehung zu folgenden Parametern:

HNDDAT0 bis 8, WRKDAT0 bis 8, HNDHOLD1 bis 8, ACCMODE, JADL

6.3.58 On Com GoSub (ON Communication Go Subroutine)

Funktion: Sprung zu einem Unterprogramm

Legt den Sprung in ein Unterprogramm fest, wenn ein Interrupt von einem Kommunikationskanal anliegt

Eingabeformat

On ☐ Com ☐ [(<Dateinummer>)] ☐ GoSub ☐ <Sprungziel>

<Dateinummer>	Legt die Nummer des Kommunikationskanals fest 1 ≤ Dateinummer ≤ 3
<Sprungziel>	Legt eine Marke fest

Programmbeispiel

Springt zu Marke RECV, wenn auf dem Kommunikationskanal Nummer 1 ein Interrupt anliegt

```
1  Open "COM1:" AS #1          'Öffnet Kommunikationskanal 1 als Datei Nr. 1
2  On Com(1) GoSub *RECV       'Springt zu Marke RECV, wenn auf dem
                               Kommunikationskanal Nummer 1 ein Interrupt anliegt
3  Com(1) On                   'Gibt den Kommunikations-Interrupt der
                               Datei Nr. 1 frei
4
   :
10                             'Liegt der Kommunikations-Interrupt der Datei
                               Nr. 1 in diesem Bereich an, so erfolgt ein
                               Sprung zur Marke RECV
11
12 Mov P1                      'Position P1 anfahren
13 Com(1) STop                 'Ignoriert Interrupts während der Bewegung von
                               P1 nach P2
14 Mov P2                      'Position P2 anfahren
15 Com(1) On                   'Sind während der Verfahrbewegung von P1 nach
                               P2 Interrupts aufgetreten, erfolgt nun deren
                               Verarbeitung
16                             'Liegt der Kommunikations-Interrupt
   :                           der Datei Nr. 1 in diesem Bereich an,
22                             so erfolgt ein
23                             Sprung zur Marke RECV
24 Com(1) Off                  'Sperrt den Kommunikations-Interrupt der
                               Datei Nr. 1
25 Close #1                    'Schließt Datei Nummer 1
26 End                         'Programmende
   :
   :
30 *RECV                       'Interruptprozedur
31 Input #1,M0001              'Speichert die empfangenen Daten in die
32 Input #1,P0001              Variablen M0001 und P0001
   :
39 Return 1                    'Springt einen Schritt hinter den Programmschritt, in dem der
                               Interrupt aufgetreten ist
```

Erläuterung

- Bei fehlender Nummer des Kommunikationskanals wird der Standardwert 1 gesetzt.
- Die Prioritäten der Interrupts werden mit steigender Dateinummer kleiner.
- Bei anliegendem Interrupt wird die Roboterbewegung aller Roboter, deren Programme in einem Programmplatz abgearbeitet werden, gestoppt. Mit der Com-Stop-Anweisung kann der Interrupt ignoriert werden und die Roboterbewegung wird nicht unterbrochen.
- In der Grundeinstellung sind die Interrupts gesperrt. Geben Sie die Interrupts nach Ausführung des Befehls über den Com-On-Befehl wieder frei.
- Der Rücksprung aus dem Unterprogramm muss mit der Return-Anweisung erfolgen. Ein Rücksprung über die GoTo-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.

Steht in Beziehung zu folgenden Befehlen:

Com On/Com Off/Com Stop, Return, Open, Input, Print, Close

6.3.59 On GoSub (ON Go Subroutine)

Funktion: Sprung zu einem Unterprogramm

Legt den Sprung zur Marke eines Unterprogramms fest

Eingabeformat

On ☐ <Ausdruck> ☐ GoSub ☐ [<Sprungziel>] [,[<Sprungziel>]] ...

<Ausdruck>	Legt fest, zu welcher Marke das Programm verzweigt wird
<Sprungziel>	Legt eine Marke fest Die maximale Anzahl beträgt 32.

Programmbeispiel

Der durch 3 Bits festgelegte Wert des Eingangssignals 16 wird in M1 übertragen. In Abhängigkeit von M1 (1 bis 7) erfolgt eine Programmverzweigung.

```
1  M1 = M_Inb(16) AND &H7      'Schreibt die Eingangssignalbits 16 bis 18
                               als 8-Bit-Wort in die numerische Variable M1
2  On M1 GoSub *L1,*LSUB,*L2,*L2,*L2,*L67,*L67   'Springt zur Marke L1,
                                                  falls M1 = 1, springt zur
                                                  Marke LSUB, falls M1 = 2,
                                                  springt zur Marke L2,
                                                  falls M1 = 3, 4 oder 5
                                                  und springt zur Marke L67,
                                                  falls M1 = 6 oder 7
10 *L1                         'Sprungmarke L1 festgelegt
11                             'Prozedur bei M1 = 1
12
13 Return                      'Rücksprung
   :
20 *LSUB                       'Sprungmarke LSUB festgelegt
21                             'Prozedur bei M1 = 2
22 Return                      'Rücksprung
   :
30 *L67                        'Sprungmarke L67 festgelegt
31                             'Prozedur bei M1 = 6 oder 7
32 Return                      'Rücksprung
   :
40 *L2                         'Sprungmarke L2 festgelegt
41                             'Prozedur bei M1 = 3, 4 oder 5
42
43 Return                      'Rücksprung
```

Erläuterung

- Der Wert des Ausdrucks legt fest, zu welcher Marke das Programm verzweigt wird.
  Beispiel: Ist der Wert des Ausdrucks 2, wird das zweite Sprungziel aufgerufen.
- Ist der Wert des Ausdrucks größer als die Anzahl der angegebenen Sprungziele, springt die Programmsteuerung zum nächsten Programmschritt.
- Wird eine Marke aufgerufen, die nicht existiert oder zweimal definiert ist, erfolgt eine Fehlermeldung.
- Der Rücksprung aus dem Unterprogramm muss mit der Return-Anweisung erfolgen. Ein Rücksprung über die GoTo-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.

Wert von <Ausdruck>	**Steuerung**
Reelle Zahl	Der Wert wird zu einer Integer-Zahl gerundet.
Wenn der Wert des Ausdrucks gleich 0 ist oder wenn der Wert größer als die Anzahl der Marken ist	Steuerung springt zum nächsten Programmschritt.
Wenn der Wert negativ oder größer als 32767 ist	ERROR
Marke ist nicht angegeben	ERROR

Tab. 6-11: *Werte des Ausdrucks und deren Verarbeitung*

6.3.60 On GoTo (ON Go To)

Funktion: Sprung zu einem Unterprogramm

Legt den Sprung zu einer Marke fest

Eingabeformat

On ☐ <Ausdruck> ☐ GoTo ☐ [<Sprungziel>] [,[<Sprungziel>]] ...

<Ausdruck>	Legt fest, zu welcher Marke das Programm verzweigt wird
<Sprungziel>	Legt eine Marke fest Die maximale Anzahl beträgt 32.

Programmbeispiel

In Abhängigkeit von M1 (1 bis 7) erfolgt eine Programmverzweigung.

```
10 On M1 GoTo *L1,*LJMP,*L2,*L2,*L67,*L67   'Springt zur Marke L1,
                                            falls M1 = 1, springt zur
                                            Marke LJMP, falls M1 = 2,
                                            springt zur Marke L2,
                                            falls M1 = 3, 4 oder 5
                                            und springt zur Marke L67,
                                            falls M1 = 6 oder 7
11                                          'Dieser Programmschritt wird ausgeführt, falls M1 keinem
                                            der Werte von 1 bis 7 entspricht (z. B. 0, 8 oder größer)
20 *L1                                      'Sprungmarke L1 festgelegt
21                                          'Prozedur bei M1 = 1
22 End                                      'Programmende
   :
30 *LJMP                                    'Sprungmarke LJMP festgelegt
31                                          'Prozedur bei M1 = 2
32 End                                      'Programmende
   :
40 *L67                                     'Sprungmarke L67 festgelegt
41                                          'Prozedur bei M1 = 6 oder 7
42 End                                      'Programmende
   :
50 *L2                                      'Sprungmarke L2 festgelegt
51                                          'Prozedur bei M1 = 3, 4 oder 5
52 End                                      'Programmende
```

Erläuterung

- Der Wert des Ausdrucks legt fest, zu welcher Marke das Programm verzweigt wird.
  Beispiel: Ist der Wert des Ausdrucks 2, wird das zweite Sprungziel aufgerufen.
- Ist der Wert des Ausdrucks größer als die Anzahl der angegebenen Sprungziele, springt die Programmsteuerung zum nächsten Programmschritt.
- Wird eine Marke aufgerufen, die nicht existiert oder zweimal definiert ist, erfolgt eine Fehlermeldung.

Wert von <Ausdruck>	**Steuerung**
Reelle Zahl	Der Wert wird zu einer Integer-Zahl gerundet.
Wenn der Wert des Ausdrucks gleich 0 ist oder wenn der Wert größer als die Anzahl der Marken ist	Steuerung springt zum nächsten Programmschritt.
Wenn der Wert negativ oder größer als 32767 ist	ERROR
Marke ist nicht angegeben	ERROR

Tab. 6-12: *Werte des Ausdrucks und deren Verarbeitung*

6.3.61 Open (Open)

Funktion: Datei öffnen

Öffnet eine Datei oder einen Kommunikationskanal

Eingabeformat

Open ☐ "<Dateibezeichnung>" ☐ [For <Modus>] ☐ AS ☐ [#]<Dateinummer>

<Dateibezeichnung> Gibt den Namen der Datei oder des Kommunikationskanals an
Die Dateibezeichnung "<Dateiname des Kommunikationskanals>:" wird zum Öffnen von Kommunikationskanälen verwendet.
Die Dateibezeichnung "<Dateiname>" wird zum Öffnen anderer Dateien verwendet.

Dateibezeichnung	Dateiname	Zugriffsmethode
Dateiname	Maximal 16 Zeichen	Input, Output, Append
Kommunikationskanal	COM 1: Standardschnittstelle RS232 (Grundeinstellung), COM 2: Einstellung des : Parameters COMDEV, COM 8: Einstellung des Parameters COMDEV	Keine Angabe = Wahlfreier Modus
	ENET: 192.168.0.2 ①	Befehl Mxt

Tab. 6-13: *Dateibezeichnung und Zugriffsmethode*

① Legen Sie hier die IP-Adresse des Kommunikationskanals fest, über den die Positionsdaten bei Verwendung des Mxt-Befehls zur externen Steuerung des Roboters übertragen werden sollen.

<Modus> Legt die Methode fest, mit der auf eine Datei zugegriffen wird
Die Angabe kann zum Öffnen von Kommunikationskanälen weggelassen werden.
- Keine Angabe = wahlfreier Modus
  Dieser Modus wird beim Zugriff auf die Kommunikationskanäle verwendet.
- INPUT = Eingabemodus
  Liest Daten von einer vorhandenen Datei ein
- OUTPUT = Ausgabemodus (neue Datei)
  Legt eine neue Datei an und schreibt Daten in diese Datei
- APPEND = Ausgabemodus (vorhandene Datei)
  Hängt Daten an das Ende einer vorhandenen Datei an

<Dateinummer> Konstante im Bereich zwischen 1 und 8;
für einen Interrupt eines Kommunikationskanals: 1 bis 3

Programmbeispiel

Öffnen eines Kommunikationskanals

```
1  Open "COM1:" AS #1          'Öffnet den RS232C-Kommunikationskanal als Datei Nr. 1
2  Mov P_01                    'Position P_01 anfahren
3  Print #1,P_CURR             'Sendet die Daten der aktuellen Position an einen an der
                               RS232 angeschlossenen Empfänger, z. B. Terminal,
                               SPS ...
                               Format: (100.00,200.00,300.00,400.00)(7.0)
4  Input #1,M1,M2,M3           'Liest die Daten „101.00,202.00,303.00“ im
                               ASCII-Format ein und schreibt sie in die Variablen
                               M1, M2 und M3
5  P_01.X = M1                 'Schreibt die Daten aus M1 in die X-Komponente
                               der globalen Variablen P_01
6  P_01.Y = M2                 'Schreibt die Daten aus M2 in die Y-Komponente
                               der globalen Variablen P_01
7  P_01.C = RAD(M3)            'Schreibt die Daten aus M3 in die C-Komponente
                               der globalen Variablen P_01
8  Close                       'Schließt alle geöffneten Dateien
9  End                         'Programmende
```

Öffnen einer Datei

```
1  Open "temp.txt" FOR APPEND AS #1   'Hängt die Daten der Datei temp.txt an das
                                      Ende der Datei Nr. 1
2  PRINT #1,"abc"                     'Schreibt die Daten „abc“ in Datei Nr. 1
3  Close #1                           'Schließt die Datei Nr. 1
```

Erläuterung

- Die Datei wird über die Dateibezeichnung geöffnet und es wird eine Dateinummer festgelegt. Schreib- oder Lesevorgänge werden einer Datei durch Angabe der Dateinummer zugeordnet.
- Ein Kommunikationskanal wird wie eine Datei behandelt.

Steht in Beziehung zu folgenden Befehlen:

Close, Print, Input, Mxt

Steht in Beziehung zu folgenden Parametern:

COMDEV

6.3.62 Ovrd (Override)

Funktion: Übersteuerung

Legt den Programmwert für die Geschwindigkeitsübersteuerung fest

Eingabeformat

Ovrd ☐ <Übersteuerungswert>

Ovrd ☐ <Übersteuerungswert> [,<Übersteuerungswert bei Aufwärtsbewegung>] [,<Übersteuerungswert bei Abwärtsbewegung>

<Übersteuerungswert>	Legt den prozentualen Übersteuerungswert fest (Standardwert: 100) 1 ≤ Übersteuerungswert ≤ 100.0 Bei einer Einstellung eines Wertes außerhalb des Einstellbereiches erfolgt eine Fehlermeldung.
<Übersteuerungswert bei Aufwärtsbewegung> <Übersteuerungswert bei Abwärtsbewegung>	Legt den Übersteuerungswert für Auf- bzw. Abwärtsbewegungen bei Bogen-Interpolation (Mva) fest.

Programmbeispiel

```
1 Ovrd 50          'Übersteuerung auf den Wert 50 % einstellen
2 Mov P1           'Position P1 mittels Gelenk-Interpolation anfahren
3 Mvs P2           'Position P2 mittels Linear-Interpolation anfahren
4 Ovrd M_NOVRD     'Standardwert einstellen
5 Mov P1           'Position P1 mittels Gelenk-Interpolation anfahren
6 Ovrd 30,10,10    'Übersteuerung für Auf- bzw. Abwärtsbewegungen bei
                   Bogen-Interpolation auf 10 % einstellen
7 Mva P3,3         'Position P3 mittels Bogen-Interpolation über
                   Bogen 3 anfahren
```

Erläuterung

- Dieser Befehl legt den prozentualen Übersteuerungswert für die Arbeitsgeschwindigkeit des Roboters fest.
- Der Ovrd-Befehl ist unabhängig von der Art der Interpolation wirksam.
- Die aktuelle Arbeitsgeschwindigkeit ergibt sich folgendermaßen:

Gelenk-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des Ovrd-Befehls	×	Einstellwert des JOvrd-Befehls
Linear-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des Ovrd-Befehls	×	Einstellwert des Spd-Befehls

- Der Maximalwert der Arbeitsgeschwindigkeit ist 100 %. Der Standardwert der Arbeitsgeschwindigkeit beträgt 100 % der Standardeinstellung (M_NOvrd).
- Der Standardwert bleibt so lange wirksam, bis der Ovrd-Befehl ausgeführt wird. Die so festgesetzte Arbeitsgeschwindigkeit kann durch einen weiteren Ovrd-Befehl geändert werden.
- Die durch einen Ovrd-Befehl festgelegte Arbeitsgeschwindigkeit bleibt so lange erhalten, bis erneut ein Ovrd-Befehl, eine End-Anweisung oder ein Reset ausgeführt wird. Nach Ausführung der End-Anweisung oder eines Resets ist der Standardwert wieder gültig.
- Liegt der Übersteuerungswert außerhalb des Wertebereiches des Roboters, erfolgt eine Fehlermeldung. Der Wert muss zwischen 0 und 100 % liegen.

Steht in Beziehung zu folgenden Befehlen:

JOvrd (für Gelenk-Interpolation), Spd (für Linear- und Kreis-Interpolation)

Steht in Beziehung zu folgenden Systemvariablen:

M_JOvrd/M_NJOvrd/M_OPOvrd/M_Ovrd/M_NOvrd
M_NOvrd (Standardeinstellung), M_Ovrd (aktuelle Einstellung)

6.3.63 Plt (Pallet)

Funktion: Koordinaten für eine Palette berechnen

Berechnet die Koordinaten eines Gitterpunktes der festgelegten Palette und weist die berechneten Koordinaten der festgelegten Position zu

Eingabeformat

Plt □ <Palettennummer>,<Gitterpunktnummer>

<Palettennummer>	Wählt eine vorher mit dem Def Plt-Befehl definierte Palette aus Die Angabe erfolgt als Konstante oder Variable. 1 ≤ Palettennummer ≤ 8
<Palettennummer>	Legt die Positionsnummer für die berechneten Koordinaten fest Die Angabe erfolgt als Konstante oder Variable.

Programmbeispiel (RV-Roboter)

```
1  Def Plt 1,P1,P2,P3,P4,4,3,1   'Definiert Palette Nummer 1
11
12 M1 = 1                        'Setzt M1 auf „1“
13  *LOOP                        'Sprungmarke LOOP festgelegt
14 Mov PICK,50                   'Position anfahren, die um 50 mm in
                                 Werkzeugzeuglängsrichtung von der
                                 Aufnahmeposition entfernt liegt
15 OVRD 50                       'Übersteuerung auf den Wert 50 % einstellen
16 Mvs PICK                      'Aufnahmeposition mittels Linear-Interpolation
                                 anfahren
17 HClose 1                      'Schließt Hand 1
18 Dly 0.5                       'Wartezeit von 0,5 Sekunden ermöglicht ein
                                 sicheres Schließen der Hand
19 Ovrd 100                      'Übersteuerung auf den Wert 100 % einstellen
20 Mvs ,50                       'Position anfahren, die 50 mm in
                                 Werkzeugzeuglängsrichtung von der aktuellen
                                 Position entfernt ist
21 PLACE = Plt 1,M1              'Weist PLACE die Koordinaten des Gitterpunktes M1 zu
22 Mov PLACE,50                  'Position anfahren, die um 50 mm in
                                 Werkzeugzeuglängsrichtung von der
                                 Ablageposition entfernt liegt
23 Ovrd 50                       'Übersteuerung auf den Wert 50 % einstellen
24 Mvs PLACE                     'Ablageposition mittels Linear-Interpolation anfahren
25 HOpen 1                       'Öffnet Hand 1
26 DLY 0.5                       'Wartezeit von 0,5 Sekunden ermöglicht ein
                                 sicheres Öffnen der Hand
27 Ovrd 100                      'Übersteuerung auf den Wert 100 % einstellen
28 Mvs ,50                       'Position anfahren, die 50 mm in
                                 Werkzeugzeuglängsrichtung von der aktuellen
                                 Position entfernt ist
29 M1 = M1 + 1                   'Erhöht den Wert von M1 um 1
30 If M1 <= 12 Then *LOOP        'Wiederholt die Schleife ab der Marke LOOP,
                                 solange M1 kleiner gleich 12 ist
31 Mov PICK,50                   'Position anfahren, die um 50 mm in
                                 Werkzeugzeuglängsrichtung von der
                                 Aufnahmeposition entfernt liegt
32 End                           'Programmende
```

Erläuterung

- Dieser Befehl berechnet die Koordinaten eines Gitterpunktes einer Palette, die vorher mit dem Def-Plt-Befehl definiert wurde, und weist sie einer Position zu.
- Die Palettennummern müssen im Bereich von 1 bis 8 liegen. Es können bis zu acht Paletten gleichzeitig definiert sein.
- Die Position des Gitterpunktes kann in Abhängigkeit der festgelegten Bewegungsrichtung (siehe Def-Plt-Befehl) unterschiedlich sein.
- Wird ein Gitterpunkt festgelegt, der außerhalb der Zeilen oder Spalten der definierten Palette liegt, erfolgt eine Fehlermeldung.
- Ist ein Palettengitterpunkt, der in einem Bewegungsbefehl als Zielposition angegeben ist, nicht in Klammern aufgeführt (richtig ist z. B. Mov (PLT 1,M1),–50), erfolgt eine Fehlermeldung.
- Eine detaillierte Beschreibung der Palettenfunktion finden Sie in Abschn. 4.4.

Steht in Beziehung zu folgenden Befehlen:

Def Plt

6.3.64 Prec (Precision)

Funktion: hohe Verfahrweggenauigkeit

Die Verfahrwegtreue bei der Ausführung von Bewegungsbefehlen kann erhöht werden.

Eingabeformat

Prec □ <On/Off>

<On/Off>	On: hohe Verfahrweggenauigkeit aktiviert Off: hohe Verfahrweggenauigkeit deaktiviert

Programmbeispiel (RV-Roboter)

```
1 Prec On          'Aktiviert die hohe Verfahrweggenauigkeit
2 Mvs P1           'Position P1 mittels Linear-Interpolation und mit hoher
                   Verfahrweggenauigkeit anfahren
3 Mvs P2           'Position P2 mittels Linear-Interpolation und mit hoher
                   Verfahrweggenauigkeit anfahren
4 Prec Off         'Deaktiviert die hohe Verfahrweggenauigkeit
5 Mov P1           'Position P1 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Ausführung eines Bewegungsbefehls mit hoher Verfahrweggenauigkeit erfolgt über den Befehl Prec On.
- Durch Aktivierung der hohen Verfahrweggenauigkeit nehmen die Beschleunigungs- und Bremszeiten und somit auch die Zykluszeiten zu.
- Die hohe Verfahrweggenauigkeit ist ab dem ersten Interpolationsbefehl nach Ausführung des Befehls Prec On gültig.
- Die hohe Verfahrweggenauigkeit wird durch Ausführung des Befehls Prec Off, der End-Anweisung oder durch Zurücksetzen des Programms deaktiviert.
- Nach Einschalten der Spannungsversorgung ist die hohe Verfahrweggenauigkeit deaktiviert.
- Im JOG-Betrieb ist die Funktion immer deaktiviert.

Steht in Beziehung zu folgenden Befehlen:

LoadSet, Mv Tune

Steht in Beziehung zu folgenden Parametern:

HNDDAT0 bis 8, WRKDAT0 bis 8

6.3.65 Print # (Print)

Funktion: Daten übertragen

Überträgt Daten in eine Datei oder über eine Kommunikationsleitung
Alle Daten werden im ASCII-Format übertragen.

Eingabeformat

Print ☐ #<Dateinummer> ☐ [,[<Ausdruck>;] ... [<Ausdruck> [;]]]

<Dateinummer>	Bezieht sich auf die im Open-Befehl festgelegte Dateinummer 1 ≤ Dateinummer ≤ 8
<Ausdruck>	Legt eine numerische Variable, eine Positionsvariable oder eine Zeichenkette fest

Programmbeispiel

```
1  Open "COM1" AS #1              'Öffnet den RS232C-Kommunikationskanal als
                                  Datei Nr. 1
2  MDATA = 150                    'Setzt MDATA auf „150“
3  Print #1,"***PRINT TEST***"    'Gibt die Zeichenkette „***PRINT TEST***“ aus
4  Print #1                       'Gibt eine Leerzeile aus
5  Print #1,"MDATA =",MDATA       'Gibt die Zeichenkette MDATA = und den Wert
                                  von MDATA aus, (150)
6  Print #1                       'Gibt eine Leerzeile aus
7  Print #1,"**********"          'Gibt die Zeichenkette „**********“ aus
8  End                            'Programmende
```

Folgendes Ergebnis wird ausgegeben:

```
***PRINT TEST***


MDATA = 150


**********
```

Erläuterung

- Fehlt eine Angabe für <Ausdruck>, wird ein „Carriage Return“ ausgegeben.
- Ausgabeformat der Daten:
  Der Platz für die Ausgabe von <Ausdruck> ist in Einheiten von 14 festgelegt. Werden bei der Ausgabe mehrere Ausdrücke angegeben, muss ein Komma zwischen den einzelnen Ausdrücken stehen.
  Bei Trennung der Ausdrücke durch Semikolons werden sie ohne Zwischenraum ausgegeben.
- Nach jeder Print-Anweisung wird ein „Carriage Return“ ausgeführt.
- Fehlt der Open-Befehl, erfolgt eine Fehlermeldung.
- Enthalten die Daten ein Anführungszeichen, erfolgt die Ausgabe der Daten bis zu diesen Anführungszeichen.

Beispiel▽

```
1  M1 = 123.5
2  P1 = (130.5,–117.2,55.1,16.2,0.0,0.0)(1,0)
```

nach Eingabe von

```
3  Print #1,"OUTPUT TEST",M1,P1
```

wird

```
OUTPUT TEST        123.5        (130.5,–117.2,55.1,16.2,0.0,0.0)(1,0)
```

ausgegeben

nach Eingabe von

```
3  Print #1,"OUTPUT TEST";M1;P1
```

wird

```
OUTPUT TEST123.5(130.5,–117.2,55.1,16.2,0.0,0.0)(1,0)
```

ausgegeben

Werden die Ausdrücke durch ein Komma oder ein Semikolon getrennt, wird kein „Carriage Return“ zugelassen. Die Ausdrücke werden in einer Zeile ausgegeben. Wird kein Komma oder Semikolon eingegeben, wird ein „Carriage Return“ zugelassen.

Nach Eingabe von

```
3  Print #1,"OUTPUT TEST",
4  Print #1,M1;
5  Print #1,P1
```

wird

```
OUTPUT TEST      123.5(130.5,–117.2,55.1,16.2,0.0,0.0)(1,0)
```

ausgegeben.

△

Steht in Beziehung zu folgenden Befehlen:

Open, Close, Input

6.3.66 Priority (Priority)

Funktion: Priorität festlegen

Die Einstellung legt die Anzahl der auszuführenden Zeilen für einen Durchgang im Multitasking-Betrieb fest.

Eingabeformat

Priority □ <Anzahl der auszuführenden Zeilen> [,<Programmplatznummer>]

<Anzahl der auszuführenden Zeilen>	Anzahl der auszuführenden Zeilen für einen Durchgang Einstellbereich: 1 bis 31
<Programmplatznummer>	Legt den Programmplatz fest 1 ≤ Programmplatz ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz (Slot) gesetzt

Programmbeispiel

Programmplatz 1 (Slot 1)

```
1 Priority 3          'Setzt die Anzahl der in einem Durchgang
                      auszuführenden Zeilen für den aktuellen Programmplatz
                      auf „3“
```

Programmplatz 2 (Slot 2)

```
1 Priority 4          'Setzt die Anzahl der in einem Durchgang
                      auszuführenden Zeilen für diesen Programmplatz
                      auf „4“
```

Erläuterung

- Programme in anderen Programmplätzen werden solange nicht ausgeführt, bis die eingestellte Anzahl von Programmschritten abgearbeitet worden ist. Im Programmbeispiel oben werden zuerst 3 Programmschritte des Programms in Programmplatz 1 und anschließend 4 Programmschritte des Programms in Programmplatz 2 ausgeführt. Nach einem Durchgang beginnt dieser Zyklus von vorne.
- Die Standardeinstellung für alle Programmplätze ist „1“. Das heißt, die Programmsteuerung springt nach Ausführung jeweils eines Programmschrittes in einem Programmplatz zum nächsten Programmplatz.
- Existiert kein Programm im angegebenen Programmplatz, erfolgt eine Fehlermeldung.
- Eine Änderung der Prioritäten ist auch dann möglich, wenn das Programm im entsprechenden Slot ausgeführt wird.

6.3.67 RelM (Release Mechanism)

Funktion: Mechanismuszuordnung aufheben

Der Befehl wird im Multitasking-Betrieb zur Steuerung von Mechanismen über Programmplätze verwendet. Er dient zur Aufhebung der über den GetM-Befehl definierten Zuordnung eines Mechanismus.

Eingabeformat

RelM

Programmbeispiel

Programmplatz 2 wird über den Programmplatz 1 gestartet. Der Mechanismus 1 im Programmplatz 2 wird über Programmplatz 1 gesteuert.

```
1  RelM                  'Definition von Mechanismus 1 aufheben, um
                         Mechanismus 1 über Programmplatz 2 zu steuern
2  XRun 2,"10"           'Startet Programm 10 als Programmplatz 2
3  Wait M_RUN(2) = 1     'Wartestatus, bis Betriebssignal des Programmplatzes 2
                         gleich 1 ist
       :
```

Programmplatz 2 (Programm 10)

```
1  GetM 1                'Definition des Mechanismus 1
2  Servo On              'Servospannung des Mechanismus 1 einschalten
3  Mov P1                'Position 1 mittels Gelenk-Interpolation anfahren
4  Mvs P2                'Position 2 mittels Linear-Interpolation anfahren
5  Servo Off             'Servospannung des Mechanismus 1 ausschalten
6  RelM                  'Definition von Mechanismus 1 aufheben
7  End                   'Programmende
```

Erläuterung

- Dieser Befehl hebt die aktuelle Zuordnung eines Mechanismus auf.
- Bei einem Programmstopp durch ein Interrupt-Signal wird der Befehl RelM automatisch durch das System ausgeführt.
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.

Steht in Beziehung zu folgenden Befehlen:

GetM

6.3.68 Rem (Remarks)

Funktion: Kommentar

Ermöglicht dem Programmierer, einen Kommentar zu schreiben

Eingabeformat

Rem ☐ [<Kommentar>]

<Kommentar>	Es können Zeichenketten bis zur Länge einer Zeile eingegeben werden.

Programmbeispiel

```
1 Rem ***Hauptprogramm***   'Legt die Zeichenkette ***Hauptprogramm***
2 '***Hauptprogramm***      als Kommentar fest
3 Mov P1                    'Position P1 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Anstelle der Rem-Anweisung kann wahlweise ein Apostroph (') verwendet werden.
- Ein Kommentar kann wie im obigen Programmbeispiel in Programmschritt 3 hinter einem Befehl in derselben Zeile stehen.

6.3.69 Reset Err (Reset Error)

Funktion: Fehler zurücksetzen

Über den Befehl kann ein vom Steuergerät generierter Fehler zurückgesetzt werden. Eine Verwendung des Befehls im Initialisierungszustand ist nicht erlaubt. Tritt außer einer Warnmeldung ein Fehler auf, können andere als kontinuierlich ausgeführte Programme nicht mehr ausgeführt werden. Die Verwendung des Befehls ist also in kontinuierlich ausgeführten Programmen sinnvoll.

Eingabeformat

Reset Err

Programmbeispiel

Befehlsausführung in einem kontinuierlich ausgeführtem Programm

```
1  If M_Err = 1 Then Reset Err    'Tritt ein vom Steuergerät generierter Fehler auf, wird der
                                  Fehler zurückgesetzt.
```

Erläuterung

- Der Befehl wird in Programmen, in denen die Startbedingung in den Programmplatzparametern auf „kontinuierliche Ausführung (ALWAYS)“ gesetzt ist, zum Zurücksetzen eines Roboter-Systemfehlers verwendet.
- Die Freigabe des Befehls erfolgt nach Einstellung des Parameters ALWENA von „0“ auf „1“ und anschließendem Aus- und Wiedereinschalten der Spannungsversorgung.

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_Err/M_ErrLvl/M_Errno

6.3.70 Return (Return)

Funktion: Rücksprung zum Hauptprogramm

Springt beim Rücksprung aus einem Unterprogramm zu dem Programmschritt nach dem GoSub-Befehl

Springt beim Rücksprung aus einer Interrupt-Routine zu dem Programmschritt zurück, in dem der Interrupt aufgetreten ist oder zum nächsten Programmschritt

Eingabeformat

Beim Rücksprung aus einem Unterprogramm:

```
Return
```

Beim Rücksprung aus einer Interrupt-Routine:

```
Return <Rücksprungziel>
```

<Rücksprungziel>	Legt den Programmschritt fest, zu der die Steuerung zurückspringt, nachdem eine Interrupt-Routine abgearbeitet wurde 0 ... Springt zu dem Programmschritt, in dem der Interrupt aufgetreten ist. 1 ... Springt einen Programmschritt hinter den Schritt, in dem der Interrupt aufgetreten ist (auch bei fehlender Angabe).

Programmbeispiel

Rücksprung aus einem Unterprogramm

```
1  '***Hauptprogramm***     'Legt die Zeichenkette „***Hauptprogramm***“ als
                            Kommentar fest
2  GoSub *SUB_INIT          'Sprung zum Unterprogramm SUB_INIT
3  Mov P1                   'Position P1 mittels Gelenk-Interpolation anfahren
   :
10 '***SUB_INIT***          'Legt die Zeichenkette „***SUB_INIT***“ als
                            Kommentar fest
11 *SUB_INIT                'Sprungmarke SUB_INIT festgelegt
12    PSTART = P1           'Weist PSTART den Wert von P1 zu
13    M100 = 123            'Weist M100 den Wert 123 zu
14 Return                   'Springt einen Programmschritt hinter den Schritt, in dem der
                            Interrupt aufgetreten ist (Programmschritt 3)
```

Rücksprung aus einer Interrupt-Routine

```
1   Def Act 1,M_In(17) = 1 GoSub *SUB1   'Definiert einen Unterprogrammsprung
                                         zur Marke SUB1, wenn der Status des
                                         allgemeinen Eingangssignals
                                         Nummer 17 = EIN ist
2   Act 1 = 1                            'Interrupt 1 freigeben
    :
10  *SUB1                                'Sprungmarke SUB1 festgelegt
                                          Interrupt-Routine für Interrupt 1
11    Act 1 = 0                          'Interrupt 1 sperren
12    M_Timer(1) = 0                     'Zähler zurücksetzen
13    Mov P2                             'Position P2 anfahren
14    Wait M_In(17) = 0                  'Wartestatus, bis Eingangsbit 17 gleich 0 ist
15    Act 1 = 1                          'Interrupt 1 erneut freigeben
16  Return 0                             'Sprung zu dem Programmschritt, aus dem der
                                         Interrupt aufgerufen wurde
```

Erläuterung

- Der Rücksprung aus einem Unterprogramm oder einer Interrupt-Routine, die mit dem Befehl GoSub aufgerufen wurde, erfolgt über die Return-Anweisung.
- Wird die Return-Anweisung ohne vorhergehende GoSub-Anweisung ausgeführt, erfolgt eine Fehlermeldung.
- Der Rücksprung aus einem mit der GoSub-Anweisung aufgerufenen Unterprogramm muss mit der Return-Anweisung erfolgen. Ein Rücksprung über die GoTo-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.
- Es erfolgt eine Fehlermeldung, wenn bei einem Return-Befehl in einem Unterprogramm ein Rücksprungziel angegeben wurde. Es erfolgt eine Fehlermeldung, wenn das Rücksprungziel in einer Interrupt-Routine nicht angegeben wurde.
- Sperren Sie den Interrupt, wenn der Rücksprung über Return 1 zu dem Programmschritt, der dem Schritt mit dem Interruptaufruf folgt, erfolgte. Wird der Interrupt nicht gesperrt und die Interrupt-Bedingung ist erfüllt, erfolgt eine erneute Ausführung der Interrupt-Routine und der Programmschritt kann beim Rücksprung übersprungen werden. Eine detaillierte Beschreibung zur Definition von Interrupt-Prozessen finden Sie in Abschn. 6.3.19.

Steht in Beziehung zu folgenden Befehlen:

GoSub, On GoSub, On Com GoSub, Def Act

6.3.71 Select Case (Select Case)

Funktion: Prozess ausführen

Führt in Abhängigkeit einer Bedingung einen von mehreren Prozessen aus

Eingabeformat

```
Select ☐ <Auswahl>
  Case ☐ <Ausdruck>
       [<Prozess>]
       Break
  Case ☐ <Ausdruck>
       [<Prozess>]
       Break

          :

  Default
       [<Prozess>]
       Break
End ☐ Select
```

<Auswahl>	Legt einen numerischen Ausdruck fest
<Ausdruck>	Legt einen Ausdruck in folgendem Format fest Der Typ muss mit dem der Bedingung übereinstimmen. – IS: <Vergleichsoperator> <Konstante> – <Konstante> – <Konstante> TO <Konstante>
<Prozess>	Legt die auszuführende Anweisung (außer GoTo-Anweisung) fest

Programmbeispiel

```
1  Select MCNT                 'Auswahl der numerischen Variablen MCNT
2      M1 = 10                 'Dieser Programmschritt wird nicht ausgeführt.
3  Case Is <= 10               'Fahre Position P1 an, falls MCNT kleiner gleich 10 ist
4      Mov P1
5      Break                   'Sprung hinter die End Select-Anweisung
6  Case 11                     'Fahre Position P2 an, falls MCNT gleich 11 oder 12 ist
7  Case 12
8      Mov P2
9      Break                   'Sprung hinter die End Select-Anweisung
10 Case 13 TO 18               'Fahre Position P4 an, falls MCNT größer gleich
                               13 und kleiner gleich 18 ist
11     Mov P4
12     Break                   'Sprung hinter die End Select-Anweisung
13 Default                     'Setzt Ausgangsbit 10 auf „1“, falls MCNT
                               keinem der oben genannten Werte oder
                               Wertebereiche entspricht
14     M_Out(10) = 1
15     Break                   'Sprung hinter die End Select-Anweisung
16 End Select
```

Erläuterung

- Wird eine der Bedingungen der Case-Anweisung erfüllt, wird der Prozess bis zur nächsten Case-, Default- oder End-Select-Anweisung ausgeführt.
- Wird keine der Case-Bedingungen erfüllt, wird der Default-Prozess ausgeführt. Ist kein Default-Prozess definiert, springt das Programm einen Programmschritt hinter die End-Select-Anweisung.
- Eine Select-Anweisung muss immer durch eine End Select-Anweisung abgeschlossen werden. Ein Sprung über die GoTo-Anweisung aus dem Case-Block der Select Case-Anweisung belegt Speicherplatz des für die Programmsteuerung reservierten Stapelspeichers. Bei einer kontinuierlichen Ausführung des Programms kann deshalb eine Fehlermeldung erfolgen.
- Bei Ausführung einer End-Select-Anweisung, der keine Select-Anweisung vorausgeht, erfolgt eine Fehlermeldung.
- Die Ausführung von While-Wend- oder For-Next-Schleifen innerhalb eines Case-Blocks ist möglich.
- Verwenden Sie Vergleichsoperatoren (<, =, > usw.) mit der Anweisung Case IS.

6.3.72 Servo (Servo)

Funktion: Servo ein-/ausschalten

Schaltet die Servospannung ein oder aus

Eingabeformat

Verwendung in einem normalen Programm:

Servo ☐ <On/Off>

Verwendung in einem Programm mit der Startbedingung „ALWAYS“:

Servo ☐ <On/Off> [,<Mechanismusnummer>]

<On/Off>	On: Servoversorgung einschalten Off: Servoversorgung ausschalten
<Mechanismusnummer>	Festlegung der Mechanismusnummer als Konstante oder Variable 1 ≤ Mechanismusnummer ≤ 3 Ist nur bei der Startbedingung „ALWAYS“ aktiv

Programmbeispiel

```
1 Servo On                  'Schaltet die Servospannung ein
2 *L2                       'Sprungmarke L2 festgelegt
3 If M_Svo <> 1 GoTo *L2    'Wartestatus, bis die Servoversorgung eingeschaltet ist
4 Spd M_NSpd                'Geschwindigkeit auf Standardwert setzen
5 Mov P1                    'Position P1 mittels Gelenk-Interpolation anfahren
6 Servo Off                 'Schaltet die Servospannung aus
```

Erläuterung

- Die Servospannung wird für alle Achsen ein- oder ausgeschaltet.
- Die Servospannung wird für alle Zusatzachsen ebenfalls ein- oder ausgeschaltet.
- In einem kontinuierlich ausgeführtem Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „1“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Systemvariablen:

M_Svo (1: EIN, 0: AUS)

Steht in Beziehung zu folgenden Parametern:

ALWENA

6.3.73 Skip (Skip)

Funktion: Sprung zum nächsten Programmschritt

Die Programmsteuerung springt zum nächsten Programmschritt.

Eingabeformat

Skip

Programmbeispiel

```
1  Mov P1 WthIf M_In(17) = 1,Skip      'Fährt Position 1 mittels Gelenk-Interpolation
                                       an und unterbricht die Roboterbewegung,
                                       wenn das Eingangsbit Nummer
                                       17 gleich 1 wird
                                       Die Programmsteuerung springt zum
                                       nächsten Programmschritt.
2  If M_SkipCq = 1 Then Hlt            'Unterbricht das Programm bei Ausführung
                                       der Skip-Anweisung
```

Erläuterung

- Dieser Befehl wird mit anderen Befehlen in Verbindung mit Wth und WthIf verwendet. Bei Ausführung des Befehls in Verbindung mit Wth oder WthIf wird die Programmabarbeitung innerhalb des Programmschritts unterbrochen und die Programmsteuerung springt zum nächsten Programmschritt. Die Ausführung einer Skip-Anweisung kann über die Roboterstatusvariable M_SkipCq geprüft werden.

Steht in Beziehung zu folgenden Systemvariablen:

M_SkipCq (1: Sprung, 0: kein Sprung)

6.3.74 Spd (Speed)

Funktion: Geschwindigkeit festlegen

Der Befehl legt die Geschwindigkeit für lineare und kreisförmige Bewegungen fest. Weiterhin ist eine Einstellung der werksseitig voreingestellten Geschwindigkeit möglich.

Eingabeformat

Spd ☐ <Geschwindigkeitswert>

Spd ☐ M_NSpd (werksseitig voreingestellter Geschwindigkeitswert)

<Geschwindigkeitswert>	Legt die Geschwindigkeit als reelle Zahl in mm/s fest

Programmbeispiel

```
1 Spd 100          'Legt die Geschwindigkeit auf 100 mm/s fest
2 Mvs P1           'Position P1 mittels Linear-Interpolation anfahren
3 Spd M_NSpd       'Setzt die Geschwindigkeit auf den Standardwert, so
                    dass die Verfahrbewegungen mit der maximal
                    möglichen Geschwindigkeit erfolgen
4 Mov P2           'Position P2 mittels Gelenk-Interpolation anfahren
5 Mov P3           'Position P3 mittels Gelenk-Interpolation anfahren
6 Ovrd 80          'Übersteuerung auf den Wert 80 % einstellen, so dass
                    bei Ausführung von Verfahrbewegungen mit der
                    maximal möglichen Geschwindigkeit kein Fehler wegen
                    Geschwindigkeitsüberschreitung auftritt.
7 Mov P4           'Position P4 mittels Gelenk-Interpolation anfahren
8 Ovrd 100         'Übersteuerung auf den Wert 100 % einstellen
```

Erläuterung

- Der Spd-Befehl ist nur bei linearen und kreisförmigen Bewegungen des Roboters wirksam.
- Der aktuelle Übersteuerungswert ergibt sich aus:

Aktueller Übersteuerungs-wert	=	Übersteuerungswert der T/B oder des Steuergeräts	×	Einstellwert des Ovrd-Befehls	×	Einstellwert des Spd-Befehls

- Bei Verwendung des Standardwerts (Grundeinstellung: 10000) verfährt der Roboter immer mit der maximal möglichen Geschwindigkeit.
- In Abhängigkeit der Roboterstellung kann es bei Verwendung des Standardwerts zu einer Fehlermeldung aufgrund einer Geschwindigkeitsüberschreitung kommen. Reduzieren Sie in diesem Fall die Geschwindigkeit dieser Verfahrbewegung, in dem Sie vor der Befehlszeile einen Ovrd-Befehl einfügen.
- Der Standardwert ist so lange gültig, bis mit dem Spd-Befehl ein neuer Wert festgelegt wird. Dieser kann durch den Spd-Befehl wieder geändert werden.
- Die über den Spd-Befehl festgelegte Geschwindigkeit wird bei der Ausführung der End-Anweisung auf den Standardwert zurückgesetzt.

Steht in Beziehung zu folgenden Systemvariablen:

M_Spd/M_NSpd/M_RSpd

6.3.75 SpdOpt (Speed Optimize)

Funktion: Geschwindigkeit optimieren

Optimiert während der Linear-Interpolation bei Annäherung an einen singulären Punkt (X = Y = 0: einer der singulären Punkte des Roboters) die Geschwindigkeit in horizontaler Richtung.

Die Funktion steht bei folgenden Software-Versionen und Robotermodellen zur Verfügung:

Software-Version:	SD-Serie: Sw1 oder später
	SQ-Serie: Rw1 oder später
Robotermodelle:	RH-3SDHR
	RH-3SQHR

Eingabeformat

SpdOpt□<On/Off>

<On/Off>	On: Optimierung der Geschwindigkeit freigeben Off: Optimierung der Geschwindigkeit sperren

Programmbeispiel

```
1 Mov P1                    'Position 1 anfahren
2 SpdOpt On                 'Optimierung der
                            Geschwindigkeit freigeben
3 Mvs P2                    'Position 2 anfahren
4 Mvs P3                    'Position 3 anfahren
5 SpdOpt Off                'Optimierung der
                            Geschwindigkeit sperren
6 Mvs P6                    'Position 6 anfahren
```

Erläuterung

- Soll die Geschwindigkeit des Werkzeugmittelpunktes während der XYZ-Interpolation bei Annäherung an den Nullpunkt (singulärer Punkt) aufrecht erhalten bleiben, so muss die J1-Achse mit einer höheren Geschwindigkeit rotieren (siehe Abb. 6-22). Dies kann in Abhängigkeit der Geschwindigkeit zu einem Fehler wegen Geschwindigkeitsüberschreitung führen. Durch die Freigabe der Funktion zur Optimierung der Geschwindigkeit wird die Geschwindigkeit automatisch reguliert und eine Fehlermeldung verhindert.
  Nähert sich zum Beispiel der Werkzeugmittelpunkt mit einer bestimmten Geschwindigkeit v dem Nullpunkt O, so würde es zu einer Geschwindigkeitsüberschreitung der J1-Achse kommen, wenn die vorgegebene Geschwindigkeit des Werkzeugmittelpunktes eingehalten werden soll. Ist die SpdOpt-Funktion freigegebenen, wird die Geschwindigkeit bei Annäherung an den Nullpunkt automatisch abgesenkt (siehe Punkt A in Abb. 6-23) und erst dann wieder auf den Wert v angehoben, wenn der Werkzeugmittelpunkt sich wieder vom Nullpunkt entfernt (siehe Punkt B in Abb. 6-23).

- Beziehung zu den Befehlen Ovrd und Spd
  Der Roboter bewegt sich mit folgender Geschwindigkeit:
  - Wenn die vorgegebene Geschwindigkeit (Ovrd- oder Spd-Befehl) niedriger ist ... Bewegung mit der vorgegebenen Geschwindigkeit
  - Wenn die vorgegebene Geschwindigkeit (Ovrd- oder Spd-Befehl) höher ist ... Bewegung mit der geringeren Geschwindigkeit

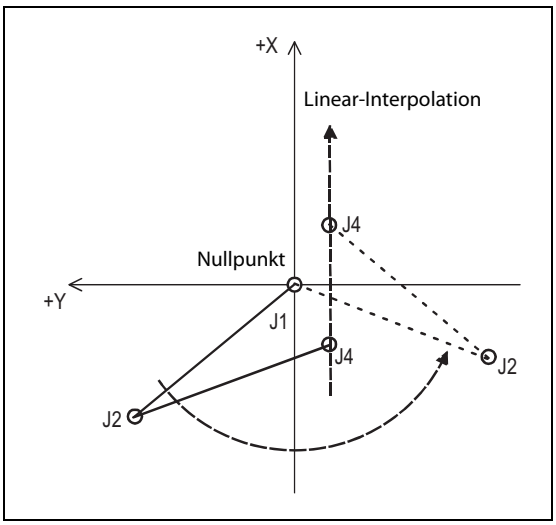


Abb. 6-22:
Annäherung an den Nullpunkt

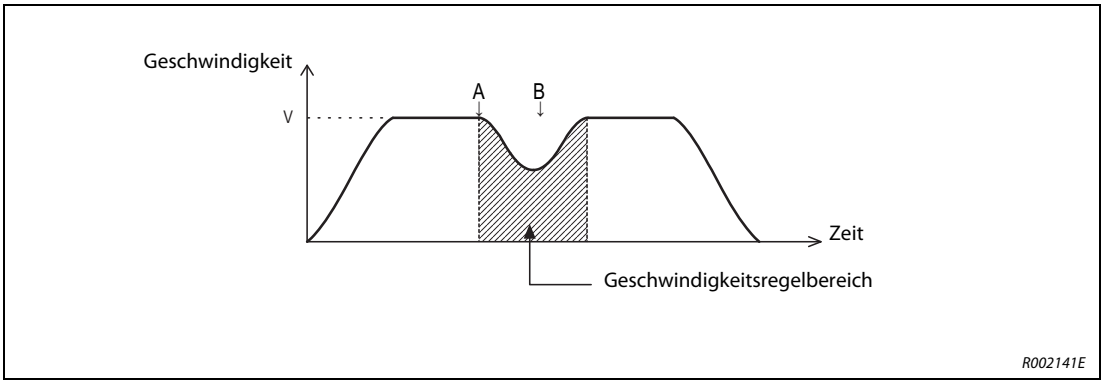


Abb. 6-23: *Beispiel zur Optimierung der Geschwindigkeit*

- Der SpdOpt-Befehl ist nur während der Linear-Interpolation wirksam. Er funktioniert nicht in der Gelenk- oder Kreisinterpolation. Die Funktion arbeitet nur dann, wenn sich die J4-Achse durch den schraffierten Bereich in Abb. 6-24 bewegt.

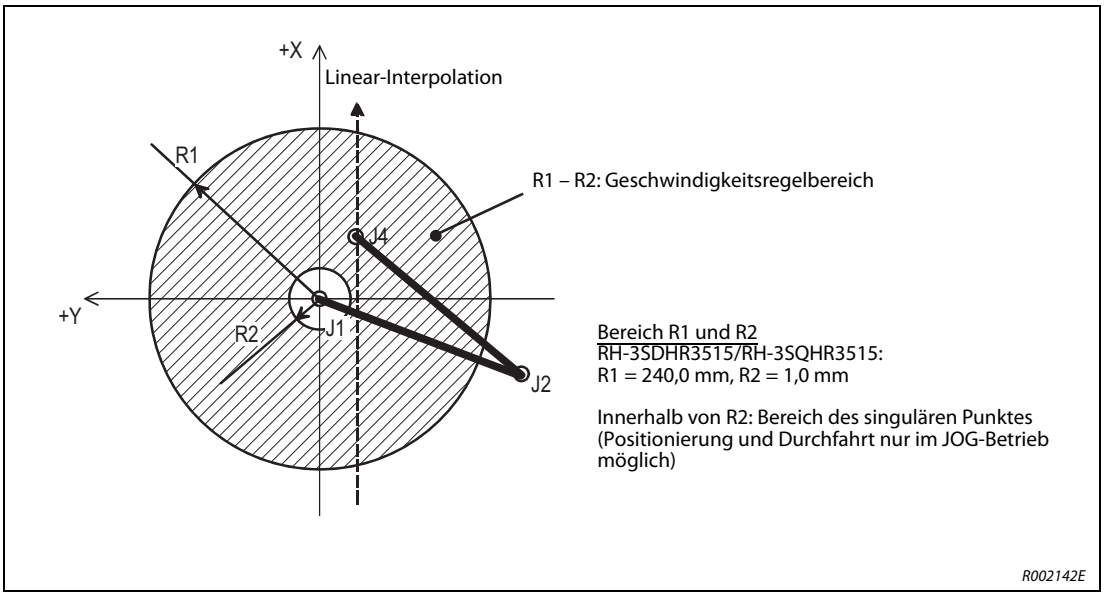


Abb. 6-24: *Arbeitsbereich des SpdOpt-Befehls und Bereich des singulären Punktes*

- Der Zustand der Funktion zur Optimierung der Geschwindigkeit nach dem Einschalten der Spannungsversorgung kann über Parameter SPDOPT festgelegt werden. Der Parameter schränkt auch die Palette der Roboter ein, mit der diese Funktion verwendet werden kann. In der Werkseinstellung ist der Parameter auf „1“ gesetzt (Optimierung der Geschwindigkeit freigegeben).
- Nach Ausführung der End-Anweisung oder nach dem Zurücksetzen des Programms ist die Einstellung gültig, die auch nach dem Einschalten der Spannungsversorgung gültig war.
- Ist die Funktion zur Optimierung der Geschwindigkeit aktiviert und die J4-Achse durchfährt den Bereich des singulären Punkts (siehe Abb. 6-24), erfolgt die Ausgabe der Fehlermeldung 2804 und der Betrieb wird unterbrochen.
- Bei Robotern, die diese Funktion nicht ausführen können, wird der SpdOpt-Befehl ignoriert, auch wenn er im Programm aufgeführt ist.
- Auch wenn die Funktion zur Optimierung der Geschwindigkeit freigegeben ist, kann ein Fehler durch Geschwindigkeitsüberschreitung auftreten, wenn bei Annäherung den singulären Punkt die Überschleiffunktion (Cnt) aktiviert ist oder bei einer Linear-Interpolation extreme Änderungen der Roboterstellung auftreten. Verwenden Sie bei aktiver Überschleiffunktion einen Verfahrweg, der weiter vom singulären Punkt entfernt ist, oder reduzieren Sie die Geschwindigkeit mit Hilfe der Ovrd-Anweisung, um das Auftreten eines Fehlers zu verhindern.
- Ist die Verfahrbewegung bei XYZ-Interpolation in horizontaler Richtung nur leicht, in vertikaler aber stark ausgeprägt, können extreme Geschwindigkeitsschwankungen auftreten, sobald die Funktion zur Optimierung der Geschwindigkeit aktiv wird. Deaktivieren Sie die Funktion in einem solchen Fall oder führen Sie die Verfahrbewegung in der Gelenk-Interpolation (Mov-Befehl) aus.

Steht in Beziehung zu folgenden Parametern:

SPDOPT

6.3.76 Title (Title)

Funktion: Programmtitel festlegen

Der Befehl legt einen Titel für ein Programm fest. Die in der Programmliste des Steuergerätes festgelegten Zeichen können über die Software RT ToolBox2 oder MELFA WORKS auf dem PC angezeigt werden.

Eingabeformat

Title □ "<Zeichenkette>"

<Zeichenkette>	Legt den Programmtitel fest

Programmbeispiel

```
1 Title "ROBOT Loader Programm"    Programmtitel „ROBOT Loader
                                   Programm“ festlegen
2 Mov P1                           'Position P1 mittels Gelenk-Interpolation
                                   anfahren
3 Mvs P2                           'Position P2 mittels Linear-Interpolation
                                   anfahren
```

Erläuterung

- Die maximale Zeichenzahl entspricht der für einen Programmschritt zulässigen Zeichenzahl. Allerdings können nur 20 Zeichen der Programmliste im Steuergerät über die Programmier-Software auf dem PC angezeigt werden.

6.3.77 Tool (Tool)

Funktion: Werkzeug-Konvertierungsdaten

Der Befehl legt die Werkzeug-Konvertierungsdaten fest (Verschiebung des TCPs).

Eingabeformat

Tool ☐ <Werkzeug-Konvertierungsdaten>

<Werkzeug-Konvertierungsdaten>	Legt die Werkzeug-Konvertierungsdaten als Positionsausdruck fest (z. B. Positionskonstanten, Positionsvariablen usw.)

Programmbeispiel

Festlegung der Werkzeug-Konvertierungsdaten als numerischer Wert

```
1 Tool (100,0,100,0,0,0)    'Der TCP wird im Werkzeugkoordinatensystem um
                            100 mm in X-Richtung und um 100 mm in Z-Richtung
                            verschoben.
2 Mvs P1                    'Position P1 mittels Linear-Interpolation anfahren
3 Tool P_NTool              'Setzt Werkzeug-Konvertierungsdaten auf den
                            Standardwert P_NTool
```

Festlegung der Werkzeug-Konvertierungsdaten als Variable

(Ist PTL01 = (100,0,100,0,0,0), entspricht dieses Beispiel dem Beispiel oben.)

```
1 Tool PLT01                'Der TCP wird im Werkzeugkoordinatensystem um
                            100 mm in X-Richtung und um 100 mm in Z-Richtung
                            verschoben.
2 Mvs P1                    'Position P1 mittels Linear-Interpolation anfahren
```

Erläuterung

- Der Befehl Tool wird z. B. beim Einsatz von Doppelgreifern zur Festlegung des Werkzeugmittelpunktes an jeder Handspitze verwendet. Sind die Werkzeugmittelpunkte beider Handgreifer identisch, sollte die Festlegung über den Parameter MEXTL und nicht über den Tool-Befehl erfolgen.
- Die mit dem Tool-Befehl geänderten Werkzeug-Konvertierungsdaten werden im Parameter MEXTL gespeichert und bleiben auch nach Ausschalten der Spannungsversorgung erhalten.
- Es wird der Standardwert (P_NTool) verwendet, bis ein Tool-Befehl ausgeführt wird. Ist der Tool-Befehl ausgeführt, sind die Werkzeug-Konvertierungsdaten so lange gültig, bis der Tool-Befehl erneut ausgeführt wird.
- Werden beim Teachen und im Automatikbetrieb unterschiedliche Werkzeug-Konvertierungsdaten verwendet, kann der Roboter nicht vorhersehbare Positionen anfahren. Stellen Sie sicher, dass die Werte bei beiden Betriebsarten übereinstimmen.
- Die für den Tool-Befehl zugelassenen Achsen sind vom Robotermodell abhängig (siehe Abschn. 9.7 „Standard-Werkzeugkoordinaten“).

Beispiel▽ Tool (0,65,145,0,0,0)

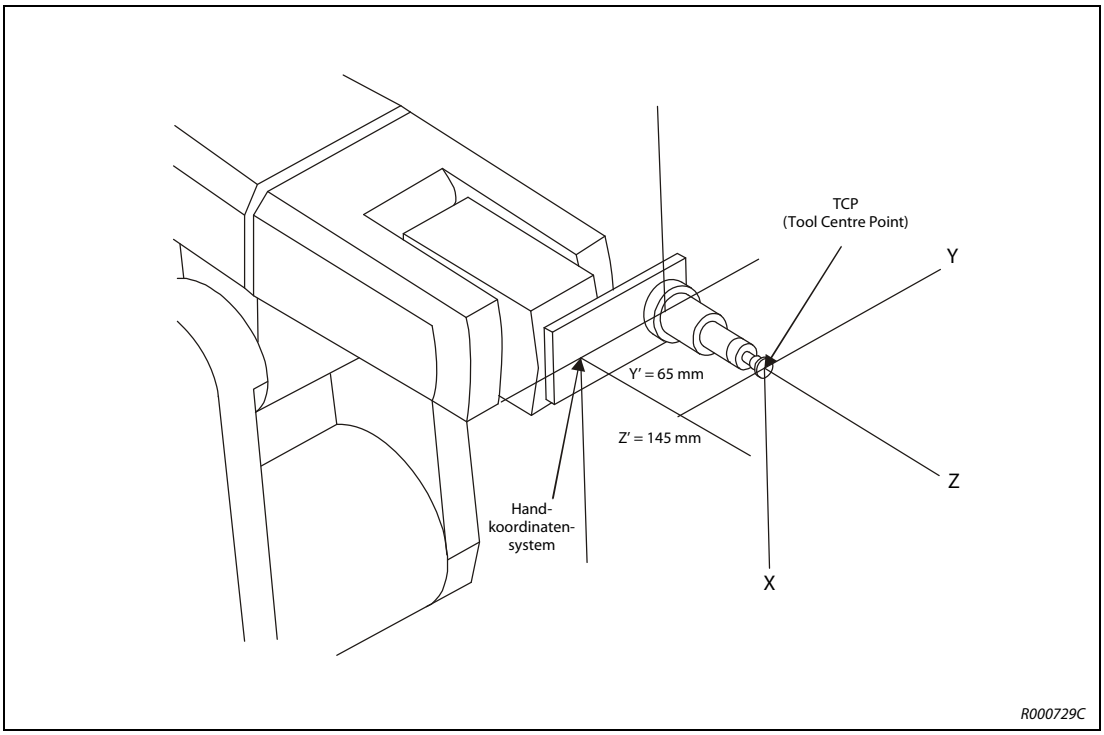


Abb. 6-25: *Übergang vom Handkoordinatensystem zum Werkzeugkoordinatensystem TCP*

△

Steht in Beziehung zu folgenden Parametern:

MEXTL, MEXTL1 bis 16 (siehe auch Abschn. 9.7)

Steht in Beziehung zu folgenden Systemvariablen:

P_NTool/P_Tool, M_Tool

6.3.78 Torq (Torque)

Funktion: Drehmomentgrenze definieren

Der Befehl legt die Drehmomentgrenze für eine Achse fest. Somit können Überlastungen vermieden werden. Beim Überschreiten der Drehmomentgrenze erfolgt die Meldung eines schweren Fehlers.

Eingabeformat

Torq □ <Achsennummer>,<Drehmomentgrenze>

<Achsennummer>	Legt die Nummer der Achse fest 1 ≤ Achsennummer ≤ 6
<Drehmomentgrenze>	Legt den prozentualen Grenzwert des Drehmoments, der von einer Achse erzeugt wird, fest 1 ≤ Grenzwert ≤ 100 %

Programmbeispiel

```
1  Def Act 1,M_Fbd > 10 GoTo *SUB1,S   'Definiert einen Unterprogrammsprung
                                       zum Programmschritt 10, wenn die
                                       Abweichung zwischen der Soll- und
                                       der Istposition größer gleich
                                       10 mm ist
2  Act 1 = 1                           'Interrupt 1 freigeben
3  Torq 3, 10                          'Legt die Drehmomentgrenze für
                                       Achse 3 auf 10 % des Maximalwerts fest
4  Mvs P1                              'Position P1 mittels Linear-Interpolation anfahren
5  Mov P2                              'Position P2 mittels Gelenk-Interpolation anfahren
6  End                                 'Programmende
10 *SUB1                               'Sprungmarke SUB1 festgelegt
11 Act 1 = 0                           'Interrupt 1 sperren
12 Mov P_Fbc                           'Soll- und Istposition abgleichen
14 M_Out(10) = 1                       'Setzt das allgemeine
                                       Ausgangssignal 10 auf „1“
15 Hlt                                 'Progamm stoppen, wenn die
                                       Abweichung zwischen Soll- und
                                       Istposition größer gleich
                                       10 mm ist
```

Erläuterung

- Es wird das maximale Drehmoment für eine Achse festgelegt. Der Grenzwert wird in Prozent, bezogen auf den Standardwert, eingestellt. Der Standardwert ist durch den Hersteller vorgegeben.
- Der Bereich der Drehmomentbegrenzung ist vom Robotermodell abhängig. Die Einstellung wird für jede Servoachse vorgenommen und entspricht somit nicht zwingend der Drehmomentgrenze an der Handspitze des Roboters. Probieren Sie verschiedene Werte aus, bis Sie die gewünschte Einstellung gefunden haben.
- Bei einem Stopp des Roboters mit aktivierter Drehmomentbegrenzung, können Abweichungen zwischen Soll- und Istposition (aufgrund von Reibung usw.) auftreten. In diesem Fall erfolgt bei der Fortsetzung des Betriebs die Meldung eines schweren Fehlers. Gleichen Sie daher die Soll- und die Istposition ab, bevor Sie den Betrieb fortsetzen (siehe Programmschritt 10 im Programmbeispiel).
- Der Befehl kann nur für Standard-Roboterachsen verwendet werden und nicht für allgemeine Servoachsen (Zusatzachsen und benutzerdefinierte Achsen). Diese Drehmomentgrenzen sind servoseitig über Parameter einzustellen

Steht in Beziehung zu folgenden Systemvariablen:

P_Fbc, M_Fbd

6.3.79 Wait (Wait)

Funktion: Wartestatus definieren

Der Befehl legt einen Wartestatus in Abhängigkeit von einer Variablen fest.

Eingabeformat

Wait □ <numerische Variable>=<numerische Konstante>

<Numerische Variable>	Legt eine numerische Variable fest Es können auch Ein- und Ausgangsvariablen (z. B. M_In, M_Out) verwendet werden.
<Numerische Konstante>	Legt eine numerische Konstante fest

Programmbeispiel

Wartestatus in Abhängigkeit eines Signals

1 Wait M_In(1) = 1	'Wartestatus, bis Eingangsbit 1 gleich 1 ist Entspricht der Befehlszeile 1 *LBL1: If M_In(1) = 0 Then GoTo *LBL1
2 Wait M_In(3) = 0	'Wartestatus, bis Eingangsbit 3 gleich 0 ist

Wartestatus in Abhängigkeit einer Anwendung

3 Wait M_Run(2) = 1	'Wartestatus, bis der Zustand des Programmplatzes 2 gleich 1 ist

Wartestatus in Abhängigkeit einer Variablen

4 Wait M_01 = 1	'Wartestatus, bis die externe Variable M_01 gleich 1 ist

Erläuterung

- Der Befehl wird zur Unterbrechung eines Programms bis zu einer Signaleingabe und während des Multitaskings verwendet.
- Der Wait-Befehl definiert einen Wartestatus. Der nächste Programmschritt wird erst dann ausgeführt, wenn die festgelegte Bedingung erfüllt ist.
- Im Multitasking-Betrieb kann die Ausführung des Wait-Befehls in mehreren Slots zu einer Verlängerung der Verarbeitungszeiten führen. Verwenden Sie in diesem Fall die If-Then-Anweisung anstelle des Wait-Befehls.
- Der Wait-Befehl darf nur eine Bedingung enthalten. Bei mehr als einer Bedingung erfolgt eine fehlerhafte Ausführung oder die Ausgabe einer Fehlermeldung.

  Nicht erlaubt wäre z. B.: Wait M_In(38) = 1 Or M_In(39) = 1
  Um die Funktion der oben genannten Befehlszeile zu erfüllen, verwenden Sie nicht den Wait-, sondern den If-Then-Else-Befehl.

Beispiel▽

```
*Loop
If M_In(38) = 1 Or M_In(39) = 1 Then *Next Else *Loop
*Next
```

6.3.80 While ~ WEnd (While End)

Funktion: Programmschleife

Solange die Schleifenbedingung „wahr" ist, wird das Programm zwischen der While- und der WEnd-Anweisung wiederholt.

Eingabeformat

```
While □ <Schleifenbedingung>
  :
WEnd
```

<Schleifenbedingung>	Legt die Abarbeitung der Schleife über eine Vergleichsbedingung fest

Programmbeispiel

```
1  While (M1 >= –5) AND (M1 <= 5)   'Wiederholt den Programmblock, solange
                                    M1 zwischen –5 und +5 liegt und springt
                                    zu Programmschritt 5, wenn M1 außerhalb des
                                    Wertebereichs liegt
2    M1 = –(M1 + 1)                 'Addiert 1 zu M1 und kehrt das Vorzeichen um
3    Print #1,M1                    'Gibt den Wert von M1 aus
4  WEnd                             'Springt zurück zur While-Anweisung
                                    (Programmschritt 1)
5  End                              'Programmende
```

Erläuterung

- Der Programmblock zwischen While und WEnd wird wiederholt, solange die Schleifenbedingung „wahr" (M1 zwischen –5 bis +5) ist.
- Ist die Schleifenbedingung „unwahr" (M1 außerhalb von –5 bis +5), springt das Programm einen Programmschritt hinter die WEnd-Anweisung.
- Ein Sprung über die GoTo-Anweisung aus einer While-WEnd-Schleife belegt Speicherplatz des für die Programmsteuerung reservierten Stapelspeichers. Bei einer kontinuierlichen Ausführung des Programms kann deshalb eine Fehlermeldung erfolgen.
  Bauen Sie das Programm so auf, dass die Schleife nur dann durchlaufen wird, wenn die Bedingung der While-Anweisung erfüllt ist.

6.3.81 Wth (With)

Funktion: Anweisung hinzufügen

Während einer Interpolationsbewegung wird eine zusätzliche Anweisung ausgeführt.

Eingabeformat

Wth ☐ <Anweisung>

<Anweisung>	Legt die zusätzlich ausgeführte Anweisung fest Es dürfen folgende Operationen ausgeführt werden: <num. Datentyp B><Substitutionsoperator><num. Datentyp A> [Substitutionen, Signal-Anweisungen (siehe entsprechendes Syntaxdiagramm)]

Programmbeispiel

```
1  Mov P1 Wth M_Out(17) = 1 Dly M1 + 2   'Position P1 anfahren und Ausgangsbit
                                         17 für die Zeit von (M1+ 2)
                                         Sekunden auf „1“ setzen
```

Erläuterung

- Dieser Befehl wird dazu verwendet, während einer Interpolationsbewegung eine zusätzliche Anweisung auszuführen.
- Es erfolgt eine Fehlermeldung, wenn die Anweisung nicht angegeben wird.
- Die Anweisung wird mit Beginn der Roboterbewegung ausgeführt.
- Die Prioritäten der Interrupts sind:
  Com > Act > WthIf (Wth)

6.3.82 WthIf (With If)

Funktion: Anweisung hinzufügen, wenn ...

Während einer Interpolationsbewegung wird eine bedingte, zusätzliche Anweisung ausgeführt.

Eingabeformat

WthIf □ <Bedingung>,<Anweisung>

<Bedingung>	Legt die Bedingung fest, bei der die zusätzliche Anweisung ausgeführt wird (siehe auch Act)
<Anweisung>	Legt die zusätzlich ausgeführte Anweisung fest (siehe auch Wth) Es dürfen folgende Operationen ausgeführt werden: – <num. Datentyp B><Substitutionsoperator><num. Datentyp A> Bsp.: M_OUT(1) = 1, P1 = P2 – Hlt-Anweisung – Skip-Anweisung

Programmbeispiel

```
1 Mov P1 WthIf M_In(17) = 1,HLT          'Position 1 anfahren
                                         und Programm stoppen,
                                         falls das Eingangsbit
                                         Nummer 17 gleich 1 ist
2 Mvs P2 WthIf M_RSpd>200,M_Out(17) = 1 Dly M1 + 2
                                         'Position 2 anfahren
                                         und das Ausgangsbit
                                         Nummer 17 für die Zeit
                                         von (M1 + 2) Sekunden
                                         auf „1“ setzen, falls die
                                         aktuelle Geschwindigkeit
                                         200 mm/s übersteigt
3 Mvs P3 WthIf M_RATIO>15,M_Out(1) = 1   'Position 3 anfahren
                                         und das Ausgangsbit
                                         Nummer 1 auf „1“
                                         setzen, sobald 15 %
                                         des Verfahrweges
                                         zurückgelegt worden
                                         sind
```

Erläuterung

- Dieser Befehl wird dazu verwendet, während einer Interpolationsbewegung eine zusätzliche, bedingte Anweisung auszuführen.
- Die Anweisung wird mit Beginn der Roboterbewegung ausgeführt.
- In der Anweisung darf kein Dly-Befehl verwendet werden.
- Beim Stoppen des Roboters über den Hlt-Befehl erfolgen die Abbremsung und der Stopp in der gleichen Weise, als wäre im Befehl Def Act die Stoppmethode 1 eingestellt (siehe auch Abschn. 6.3.19 „Def Act (Define Act)“.

6.3.83 XClr (X Clear)

Funktion: Programmauswahl zurücksetzen

Der Befehl setzt die Auswahl eines Programms in einen festgelegten Programmplatz (Slot/Task) während des Multitasking-Betriebs zurück.

Eingabeformat

XClr ☐ <Programmplatznummer>

<Programmplatznummer>	Legt die Nummer des Programmplatzes als Konstante oder Variable fest 1 ≤ Programmplatznummer ≤ 32

Programmbeispiel

```
1  XRun 2,"1"                ’Auswahl des Programms 1 für Programmplatz 2
   :
10 XStp 2                    ’Stoppt Programmplatz 2
11 Wait M_Wai(2) = 1         ’Wartestatus, bis Programmplatz 2 gestoppt ist
12 XRst 2                    ’Wartestatus von Programmplatz 2 zurücksetzen
   :
20 XClr 2                    ’Auswahl von Programmplatz 2 zurücksetzen
21 End                       ’Programmende
```

Erläuterung

- Entspricht die Programmplatznummer nicht dem ausgewählten Programmplatz, erfolgt eine Fehlermeldung.
- Wird das ausgewählte Programm während der Ausführung zurückgesetzt, erfolgt eine Fehlermeldung.
- Wird ein Programm zurückgesetzt, das sich im Wartestatus befindet, erfolgt eine Fehlermeldung.
- Die Festlegung des Programmnamens erfolgt in Anführungszeichen. In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „1“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Befehlen:

XLoad, XRst, XRun, XStp

Steht in Beziehung zu folgenden Parametern:

ALWENA

6.3.84 XLoad (X Load)

Funktion: Programm laden

Der Befehl lädt im Multitasking-Betrieb ein Programm in einen festgelegten Programmplatz (Slot/ Task).

Eingabeformat

XLoad □ <Programmplatznummer> ,"<Programmname>"

<Programmplatznummer>	Legt die Nummer des Programmplatzes als Konstante oder Variable fest 1 ≤ Programmplatznummer ≤ 32
<Programmname>	Legt den Namen des Programms fest

Programmbeispiel

```
1  If M_Psa(2) = 0 Then *L1          'Sprung zur Marke L1, falls die Programmwählbarkeit
                                     des Programmplatzes 2 gesperrt ist
2  XLoad 2,"10"                      'Auswahl des Programms 10 für
                                     Programmplatz 2
3  *L3                               'Marke L3 festgelegt
4  If C_Prg(2) <> "10" Then GoTo *L3 'Wartezeit bis das Programm gestartet wird
5  XRun 2                            'Startet Programmplatz 2
6  Wait M_Run(2) = 1                 'Wartestatus, bis das Betriebssignal des
                                     Programmplatzes 2 gleich 1 ist
7  *L1                               'Marke L1 festgelegt
8                                    'Ist der Programmplatz 2 bereits aktiv,
                                     startet die Programmausführung hier
```

Erläuterung

- Es erfolgt eine Fehlermeldung, wenn das ausgewählte Programm nicht existiert.
- Ist das gewählte Programm bereits einem anderen Programmplatz zugewiesen, erfolgt bei der Programmausführung eine Fehlermeldung.
- Ist das gewählte Programm editiert worden, erfolgt bei der Programmausführung eine Fehlermeldung.
- Wird das gewählte Programm bereits ausgeführt, erfolgt bei der erneuten Programmausführung eine Fehlermeldung.
- Die Festlegung des Programmnamens erfolgt in Anführungszeichen.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn der Parameter ALWENA von „0“ auf „1“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.
- Wird der Befehl XRun unmittelbar nach dem Befehl XLoad ausgeführt, kann ein Fehler auftreten, da der Ladevorgang des Programms noch nicht abgeschlossen ist. Prüfen Sie daher den Abschluss des Ladevorgangs (siehe Programmschritte 3 und 4 im Programmbeispiel).

Steht in Beziehung zu folgenden Befehlen:

XClr, XRst, XRun, XStp

Steht in Beziehung zu folgenden Parametern:

ALWENA

6.3.85 XRst (X Reset)

Funktion: Programm zurücksetzen

Der Befehl setzt die Steuerung des Programms des festgelegten Programmplatzes im Multitasking-Betrieb vom aktuellen Programmschritt auf den Programmanfang zurück.

Eingabeformat

XRst ☐ <Programmplatznummer>

<Programmplatznummer>	Legt die Nummer des Programmplatzes als Konstante oder Variable fest 1 ≤ Programmplatznummer ≤ 32

Programmbeispiel

```
1  XRun 2                    'Startet Programmplatz 2
2  Wait M_Run(2) = 1         'Wartestatus, bis Betriebssignal des Programmplatzes 2
   :                         gleich 1 ist
   :
10 XStp 2                    'Stoppt Programmplatz 2
11 Wait M_Wai(2) = 1         'Wartestatus, bis Programmplatz 2 gestoppt ist
   :
15 XRst 2                    'Setzt Programmplatz 2 zurück
16 Wait M_Psa(2) = 1         'Wartestatus, bis Programmplatz 2 gewählt wurde
   :
20 XRun 2                    'Startet Programmplatz 2
21 Wait M_Run(2) = 1         'Wartestatus, bis Betriebssignal des Programmplatzes 2
   :                         gleich 1 ist
```

Erläuterung

- Ein Zurücksetzen des Programms ist nur im gestoppten Zustand des Programmplatzes möglich.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn der Parameter ALWENA von „0“ auf „1“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Befehlen:

XClr, XLoad, XRun, XStp

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_Psa(Programmplatznummer) (1 = Auswahl freigegeben, 0 = Auswahl gesperrt)
M_Run(Programmplatznummer) (1 = Betrieb, 0 = kein Betrieb)
M_Wai(Programmplatznummer) (1 = Pause, 0 = keine Pause)

6.3.86 XRun (X Run)

Funktion: Programm starten

Der Befehl startet die parallele Ausführung der gewählten Programme im Multitasking-Betrieb.

Eingabeformat

XRun ☐ <Programmplatznummer>,"<Programmname>" [,<Ausführung>]

<Programmplatznummer>	Legt die Nummer des Programmplatzes als Konstante oder Variable fest 1 ≤ Programmplatznummer ≤ 32
<Programmname>	Legt den Namen des Programms fest
<Ausführung>	Legt die Ausführung des Programms fest 0 = kontinuierlich, 1 = zyklisch

Programmbeispiel

Wahl des auszuführenden Programms über den Befehl XRun (kontinuierliche Ausführung)

```
1  XRun 2,"1"                    'Startet Programm 1 als Programmplatz 2
2  Wait M_Run(2) = 1             'Wartestatus, bis Betriebssignal des
                                 Programmplatzes 2 gleich 1 ist
```

Wahl des auszuführenden Programms über den Befehl XRun (zyklische Ausführung)

```
1  XRun 3,"2",1                  'Startet Programm 2 als Programmplatz 3 im
                                 zyklischen Betrieb
2  Wait M_Run(3) = 1             Wartestatus, bis Betriebssignal des
                                 Programmplatzes 3 gleich 1 ist
```

Wahl des auszuführenden Programms über den Befehl XLoad (kontinuierliche Ausführung)

```
1  XLoad 2,"1"                   'Auswahl des Programms 1 für
                                 Programmplatz 2
2  *L2                           'Marke L2 festgelegt
3  If C_Prg(2) <> "1" Then GoTo *L2   'Wartezeit bis das Programm gestartet wird
4  XRun 2                        'Startet Programmplatz 2
```

Wahl des auszuführenden Programms über den Befehl XLoad (zyklische Ausführung)

```
1  XLoad 3,"2"                   'Auswahl des Programms 2 für
                                 Programmplatz 3
2  *L2                           'Marke L2 festgelegt
3  If C_Prg(3) <> "1" Then GoTo *L2   'Wartezeit bis das Programm gestartet wird
4  XRun 3, ,1                    'Startet Programmplatz 3 im zyklischen
                                 Betrieb
```

Erläuterung

- Es erfolgt eine Fehlermeldung, wenn das ausgewählte Programm nicht existiert.
- Wird der gewählte Programmplatz bereits verwendet, erfolgt bei der Programmausführung eine Fehlermeldung.
- Ist ein Programm noch in keinen Programmplatz geladen, erfolgt das Laden über diesen Befehl. Der Befehl XLoad kann dann entfallen.
- Wird der XRun-Befehl im Wartestatus eines in der Programmmitte gestoppten Programms ausgeführt, startet der Betrieb im kontinuierlichen Modus.
- Die Festlegung des Programmnamens erfolgt in Anführungszeichen.
- Fehlt die Angabe für die Ausführung, wird das Programm im aktuellen Modus ausgeführt.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn der Parameter ALWENA von „0“ auf „1“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.
- Wird der Befehl XRun unmittelbar nach dem Befehl XLoad ausgeführt, kann ein Fehler auftreten, da der Ladevorgang des Programms noch nicht abgeschlossen ist. Prüfen Sie daher den Abschluss des Ladevorgangs (siehe Programmschritt 3 im 3ten und 4ten Programmbeispiel).

Steht in Beziehung zu folgenden Befehlen:

XClr, XLoad, XRst, XStp

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_Run(Programmplatznummer) (1 = Betrieb, 0 = kein Betrieb)

6.3.87 XStp (X Stop)

Funktion: Programm stoppen

Der Befehl unterbricht die Ausführung des Programms des gewählten Programmplatzes (Slot/Task) im Multitasking-Betrieb. Eine über den Programmplatz ausgeführte Verfahrbewegung des Roboters wird gestoppt.

Eingabeformat

XStp □ <Programmplatznummer>

<Programmplatznummer>	Legt die Nummer des Programmplatzes als Konstante oder Variable fest 1 ≤ Programmplatznummer ≤ 32

Programmbeispiel

```
1  XRun 2                    'Startet Programmplatz 2
   :
10 XStp 2                    'Stoppt Programmplatz 2
11 Wait M_Wai(2) = 1         'Wartestatus, bis Programmplatz 2 gestoppt ist
   :
20 XRun 2                    'Startet Programmplatz 2
```

Erläuterung

- Wird ein bereits gestoppter Programmplatz gestoppt, erfolgt keine Fehlermeldung.
- Der XStp-Befehl stoppt auch ein kontinuierlich ausgeführtes Programm.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn der Parameter ALWENA von „0“ auf „1“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Befehlen:

XClr, XLoad, XRst, XRun

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_Wai(Programmplatznummer) (1 = Pause, 0 = keine Pause)

6.3.88 Substitute (Substitute)

Funktion: Daten ersetzen

Das Ergebnis einer Operation wird in eine Variable oder Feldvariable übertragen.

Eingabeformat 1

<Variablenname> = <Ausdruck 1>

Eingabeformat 2

<Variablenname> = <Ausdruck 1> Dly <Ausdruck 2>

<Variablenname>	Legt den Namen der Variablen fest, in die die Daten übertragen werden (siehe auch Syntaxdiagramme der Variablentypen)
<Ausdruck 1>	Daten, die in die Variable übertragen werden
<Ausdruck 2>	Legt die Verzögerungszeit fest

Programmbeispiel

```
1  P100 = P1 + P2 * 2      'Überträgt das Ergebnis der Operation P1 + P2 * 2 in die Variable P100
2  M_Out(10) = 1           'Setzt Ausgang 10 auf „1“
3  M_Out(17) = 1 Dly 2.0   'Setzt Ausgangsbit Nummer 17 für 2 Sekunden auf „1“
```

Erläuterung

- Wird der Befehl für einen Impulsausgang verwendet, erfolgt die Ausgabe des Impulses parallel zu den Befehlen der nachfolgenden Programmschritte.
- Bei einer Impulsausgabe über die Variablen M_Outb oder M_Outw werden die Bits in Gruppen von 8 oder 16 Bits invertiert. Es kann nicht jede beliebige Bitbreite ausgegeben werden.
- Wird während der festgesetzten Zeit eine End-Anweisung, die letzte Zeile des Programms oder ein NOT-HALT ausgeführt, behält der Impulsausgang seinen gegenwärtigen Zustand bei. Nach der abgelaufenen Verzögerungszeit werden die Bits wieder invertiert.

7 Roboterstatusvariablen

7.1 Allgemeine Hinweise

In den nachfolgenden Abschnitten finden Sie eine alphabetische Auflistung aller Roboterstatusvariablen und deren Anwendungsmöglichkeiten.

7.1.1 Beschreibung des verwendeten Formats

Funktion

Hier finden Sie eine Funktionsbeschreibung der Variablen.

Eingabeformat

Hier finden Sie das genaue Format zur Eingabe der Variablen. Die eckigen Klammern „[ ]“ kennzeichnen die wahlfreien Variablenwerte.

Systemstatusvariablen können in Vergleichsoperationen, als Bezugsparameter oder in Zuweisungsanweisungen verwendet werden. Die hier gezeigten Beispiele beschränken sich auf Anwendungen als Bezugsparameter oder Zuweisungsanweisungen.

Programmbeispiel

Hier finden Sie die Verwendung der Variablen in einem Beispielprogramm.

Erläuterung

Hier finden Sie eine detaillierte Beschreibung, Besonderheiten usw. der Variablen.

7.2 Detaillierte Variablenbeschreibung

In diesem Abschnitt finden Sie eine detaillierte Beschreibung sowie Programmbeispiele zur Anwendung der Roboterstatusvariablen.

7.2.1 C_Com

Funktion: Parametereinstellung für Kommunikationskanal

Hiermit werden die Parameter für den Kommunikationskanal eingestellt, der mit der Open-Anweisung geöffnet werden soll. Diese Variable wird eingesetzt, wenn das Ziel einer Kommunikation häufig geändert wird.

- Format Zeichenkette
- Nur für den Client im Ethernet

Eingabeformat

C_Com (<Kommunikationskanalnr.>) = "ETH: <Server-IP-Adresse> [, <Schnittstellennr.>]"

ETH:	Kennzeichnet des Ziel der Kommunikation als Ethernet
<Kommunikationskanalnr.>	Legt die Nummer der COM-Schnittstelle für die Open-Anweisung fest (Der Kanaltyp wird über COMDEV-Parameter zugeordnet.) Einstellung von 1 bis 8
<Server-IP-Adresse>	IP-Adresse des Servers (kann weggelassen werden)
<Schnittstellennr.>	Schnittstellennummer des Servers (Wird keine Schnittstellennummer eingestellt, wird die Parametereinstellung von NETPORT verwendet.)

Programmbeispiel

```
1  C_Com(2) = "ETH:192.168.0.10,10010"   'Einstellen der IP-Adresse des Ziel-Servers für die Kommunikation
                                         Kommunikationskanal ist COM2
2  *O1                                   'Sprungmarke O1 festgelegt
3  Open "COM2:" AS #1                    'Öffnen des Kommunikationskanals, nach Einstellung der IP-Adresse auf 192.168.0.10 und der Schnittstellennummer auf 10010.
4  If M_Open(1) <> 1 Then *O1            'Wiederholung der Open-Anweisung, falls keine Verbindung zum Server möglich
5  Print #1, "HELLO"                     'Senden einer Zeichenkette
6  Input #1, C1$                         'Empfang einer Zeichenkette
7  Close #1                              'Schließen des Kommunikationskanals
8  C_Com(2) = "ETH:192.168.0.11,10011"   'Einstellen der IP-Adresse des Ziel-Servers für die Kommunikation – Kommunikationskanal ist COM2
9  *O2                                   'Sprungmarke O2 festgelegt
10 Open "COM2:" AS #1                    'Öffnen des Kommunikationskanals, nach Einstellung der IP-Adresse auf 192.168.0.11 und der Schnittstellennummer auf 10011.
11 If M_Open(1) <> 1 Then *O2            'Wiederholung der Open-Anweisung, falls keine Verbindung zum Server möglich
12 Print #1, C1$                         'Senden einer Zeichenkette
13 Input #1, C2$                         'Empfang einer Zeichenkette
14 Close #1                              'Schließen des Kommunikationskanals
15 Hlt                                   'Stoppt das Programm
16 End                                   'Ende
```

Erläuterung

- Die Verwendung dieser Variable ist nicht notwendig, wenn die Einstellung des Steuergerätes zur Kommunikation mit externen Einheiten über die Parameter von NETHSTIP und NETPORT erfolgte und diese Einstellung nicht geändert werden soll.
- Diese Variable ist nur für den Client einer Datenverbindung mit der Ethernet-Option gültig.
- Da die Kommunikationsparameter für die Open-Anweisung bereits eingestellt sind, muss diese Variable vor der Open-Anweisung ausgeführt werden.
- Die Einstellungen der Parameter von NETHSTIP und NETPORT werden bei Einschalten der Spannungsversorgung verwendet. Durch die Ausführung dieser Variable werden die entsprechenden Einstellungen zeitweise geändert. Sie sind solange gültig, bis die Spannungsversorgung wieder abgeschaltet wird. Nach dem Wiedereinschalten der Spannungsversorgung werden die ursprünglichen Parametereinstellungen wieder verwendet.
- Wird diese Variable hinter der Open-Anweisung verwendet, ändert sich die Einstellung des Kommunikationskanals nicht. In diesem Fall muss der Kommunikationskanal zuerst mit der Close-Anweisung geschlossen werden, danach die Variable C_Com ausgeführt werden und dann erneut die Open-Anweisung.
- Bei Verwendung einer fehlerhaften Syntax erscheint die Fehlermeldung erst beim Ausführen des Programms und nicht beim Editieren.

Steht in Beziehung zu folgenden Parametern:

NETHSTIP, NETPORT

7.2.2 C_Date

FunktionTeaching Box: Datumseinstellungen lesen

Die Variable enthält die aktuellen Datumseinstellungen im Format Jahr/Monat/Tag.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_Date

Programmbeispiel

```
1 C1$ = C_Date          'Überträgt das aktuelle Datum, z. B. 2005/12/01 in die
                        Zeichenkettenvariable C1$
```

Erläuterung

- Die aktuellen Datumseinstellungen werden übertragen.
- Die Variable kann ausschließlich gelesen werden. Eine Änderung der Datumseinstellungen erfolgt über die Teaching Box.

Steht in Beziehung zu folgenden Variablen:

C_Time

7.2.3 C_Maker

Funktion: Herstellerangaben lesen

Die Variable enthält Herstellerangaben des Roboter-Steuergerätes.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_Maker

Programmbeispiel

```
1 C1$ = C_Maker         'Überträgt z. B. „COPYRIGHT1999.....“ in die
                        Zeichenkettenvariable C1$
```

Erläuterung

- Es werden Herstellerangaben des Roboter-Steuergeräts übertragen.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

C_Mecha

7.2.4 C_Mecha

Funktion: Mechanismusangaben lesen

Die Variable enthält die Bezeichnung des verwendeten Mechanismus.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_Mecha [(<Mechanismusnummer>)]

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1 C1$ = C_Mecha          'Falls der Robotertyp „RV-6SD“ ist, wird „RV-6SD“ in die
                         Zeichenkettenvariable C1$ übertragen.
```

Erläuterung

- Der aktuelle Mechanismustyp wird übertragen.
- Die Variable kann ausschließlich gelesen werden.

7.2.5 C_Prg

Funktion: Programmnamen lesen

Die Variable enthält die Programmnummer des ausgewählten Programms.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_Prg [(<Numerischer Ausdruck>)]

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1 C1$ = C_Prg(1)         'Falls die Programmnummer im Programmplatz 1 „10“
                         ist, wird der Wert „10“ in die Zeichenkettenvariable C1$
                         übertragen
```

Erläuterung

- Die Programmnummer des ausgewähltem Programmplatzes wird übertragen.
- Im Einzelprogrammplatzbetrieb wird die Programmplatznummer auf „1“ gesetzt.
- Bei einer Auswahl des Programms über das Steuergerät wird dieser Wert gesetzt.
- Die Variable kann ausschließlich gelesen werden.
- Ist im ausgewählten Programmplatz kein Programm geladen, erfolgt bei Ausführung der Anweisung eine Fehlermeldung.

7.2.6 C_Time

Funktion: Zeiteinstellungen lesen

Die Variable enthält die aktuellen Zeiteinstellungen in Stunden/Minuten/Sekunden im 24-Stunden-Format.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_Time

Programmbeispiel

```
1 C1$ = C_Time                    'Überträgt die aktuelle Zeit, z. B. 01/05/20 in die
                                  Zeichenkettenvariable C1$
```

Erläuterung

- Die aktuellen Zeiteinstellungen werden übertragen.
- Die Variable kann ausschließlich gelesen werden. Eine Änderung der Datumseinstellungen erfolgt über die Teaching Box.

Steht in Beziehung zu folgenden Variablen:

C_Date

7.2.7 C_User

Funktion: Benutzerinformation lesen

Die Variable enthält die Zeichen des Parameters USERMSG.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_User

Programmbeispiel

```
1 C1$ = C_User                    'Überträgt die Zeichen des Parameters USERMSG in die
                                  Zeichenkettenvariable C1$
```

Erläuterung

- Die Zeichen des Parameters USERMSG werden übertragen.
- Die Variable kann ausschließlich gelesen werden.
- Eine Änderung des Parameters erfolgt über die Software RT ToolBox2 oder die Teaching Box.

7.2.8 J_ColMxl

Funktion: Drehmomentabweichung lesen

Die Variable enthält die maximale Drehmomentabweichung zwischen dem Drehmoment-Istwert und dem Drehmoment-Sollwert bei aktivierter Kollisionsüberwachung.

Eingabeformat

Bsp.: <Gelenkvariable> = J_ColMxl [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest Auch bei einer Impulskette wird eine Gelenkvariable verwendet.
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  M1 = 100                          'Initialisierungswert der Ansprechschwelle für die
                                     Kollisionsüberwachung für alle Achsen vorgeben
2  M2 = 100
3  M3 = 100
4  M4 = 100
5  M5 = 100
6  M6 = 100
7  *Lbl1                             'Sprungmarke Lbl1 festgelegt
8  ColLvl M1,M2,M3,M4,M5,M6,,        'Ansprechschwelle für die Kollisionsüberwachung
                                     für alle Achsen einstellen
9  ColChk ON                         'Kollisionsüberwachung aktivieren
                                     (Berechnung der maximalen Drehmomentab-
                                     weichung starten)
10 MOV P1                            'Position P1 mittels Gelenk-Interpolation anfahren
   :
51 ColChk OFF                        'Kollisionsüberwachung deaktivieren
                                     (Berechnung der maximalen Drehmoment-
                                     abweichung stoppen)
52 M1 = J_ColMxl(1).J1 + 10          'Zulässige Ansprechschwelle mit einer Abweichung
                                     von 10 % für jede Achse berechnen
                                     (10 % ist ein Beispielwert für das Programmbeispiel
                                     und kein allgemeingültiger Wert.)
53 M2 = J_ColMxl(1).J2 + 10
54 M3 = J_ColMxl(1).J3 + 10
55 M4 = J_ColMxl(1).J4 + 10
56 M5 = J_ColMxl(1).J5 + 10
57 M6 = J_ColMxl(1).J6 + 10
58 GoTo *Lbl1                        'Sprung zur Marke Lbl1
```

Erläuterung

- Die maximale Drehmomentabweichung zwischen dem Drehmoment-Istwert und dem Drehmoment-Sollwert bei aktivierter Kollisionsüberwachung wird in der Variablen J_ColMxl gespeichert.

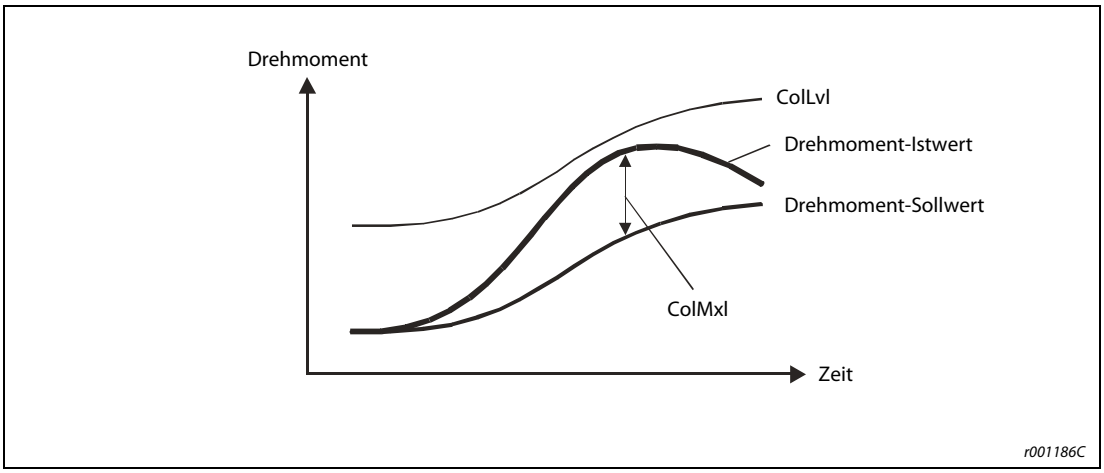


Abb. 7-1: *Drehmomentabweichung zwischen Drehmoment-Istwert und -Sollwert*

- Ein Wert der Variablen von 100 % entspricht dem zulässigen Wert der Drehmomentabweichung in der Werkseinstellung.
- Bei Robotern, die nicht über die Funktion der Kollisionsüberwachung verfügen, wird der Wert für jede Achse auf „0.0“ gesetzt.
- Wird die Servoversorgung während der Ausführung der Anweisung ColChk ON oder COLLVL eingeschaltet, erfolgt ein Zurücksetzen der Variablen auf den Wert „0“.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Befehlen:

ColChk, ColLvl

Steht in Beziehung zu folgenden Variablen:

M_ColSts, P_ColDir

7.2.9 J_Curr

Funktion: Gelenkdaten lesen

Die Variable enthält die Gelenkdaten der aktuellen Position.

Eingabeformat

Bsp.: <Gelenkvariable> = J_Curr [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

1 J1 = J_Curr	'Überträgt die Gelenkdaten der aktuellen Position in die Gelenkvariable J1

Erläuterung

- Die Gelenkdaten der aktuellen Position des festgelegten Roboters werden übertragen.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

P_Curr

7.2.10 J_ECurr

Funktion: Encoderdaten lesen

Die Variable enthält die Zahl der Encoderimpulse.

Eingabeformat

Bsp.: <Gelenkvariable> = J_ECurr [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

1 J1 = J_ECurr(1)	'Überträgt die Anzahl der Encoderimpulse in die Gelenkvariable J1
2 MA = J1.J1	'Überträgt die Anzahl der Encoderimpulse der Achse J1 in die Variable MA

Erläuterung

- Die Anzahl der Encoderimpulse wird in eine Gelenkvariable übertragen. Legen Sie dann eine Achse fest und übertragen Sie die Encoderdaten in eine numerische Variable.
- Die Variable kann ausschließlich gelesen werden.

7.2.11 J_Fbc/J_AmpFbc

Funktion: Gelenkdaten lesen

J_Fbc: Die Variable enthält Gelenkdaten der aus den Servorückmeldeimpulsen abgeleiteten aktuellen Position.
J_AmpFbc: Die Variable enthält den aktuellen Wert der Servorückmeldeimpulse jeder Achse.

Eingabeformat

Bsp.: <Gelenkvariable> = J_Fbc [(<Mechanismusnummer>)]

Bsp.: <Gelenkvariable> = J_AmpFbc [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

1 J1 = J_Fbc	'Überträgt die Gelenkdaten der aktuellen Position, die aus den Servorückmeldeimpulsen ermittelt wurden, in die Gelenkvariable J1
2 J2 = J_AmpFbc	'Überträgt den aktuellen Wert der Servorückmeldeimpulse in die Gelenkvariable J2

Erläuterung

- Die Variable J_Fbc enthält die Gelenkdaten der aus den Encoderimpulsen ermittelten aktuellen Position des festgelegten Mechanismus.
- Die Variable J_Fbc ermöglicht eine Prüfung der Verzögerungszeit zwischen den an die Servos übermittelten Steuerimpulsen und der eigentlichen Servoaktivität.
- Die Variable J_Fbc ermöglicht die Erkennung einer durch die Anweisung Cmp Jnt hervorgerufenen Abweichung.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

P_Fbc

7.2.12 J_Origin

Funktion: Grundpositionsdaten lesen

Die Variable enthält die Gelenkdaten der Grundposition (Nullpunkt).

Eingabeformat

Bsp.: <Gelenkvariable> = J_Origin [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

1 J1 = J_Origin	'Überträgt die Gelenkdaten der Grundposition in die Gelenkvariable J1

Erläuterung

- Die Gelenkdaten der eingestellten Grundposition werden übertragen.
- Die Variable ermöglicht eine Prüfung der Grundposition, z. B. bei Positionsabweichungen.
- Die Variable kann ausschließlich gelesen werden.

7.2.13 M_Acl/M_DAcl/M_NAcl/M_NDAcl/M_AclSts

Funktion: Beschleunigungs-/Abbremszeiten lesen

Die Variable enthält für die Beschleunigung/Abbremsung relevante Daten.

M_Acl: aktuelle Beschleunigungszeit (%)
M_DAcl: aktuelle Abbremszeit (%)
M_NAcl: Standardwert der Beschleunigungszeit (100 %)
M_NDAcl: Standardwert der Abbremszeit (100 %)
M_AclSts: aktueller Status der Beschleunigung/Abbremsung
0 = gestoppt
1 = beschleunigt
2 = konstante Geschwindigkeit
3 = bremst

Eingabeformat

Bsp.: <Numerische Variable> = M_Acl [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_DAcl [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_NAcl [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_NDAcl [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_AclSts [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1 M1 = M_Acl          'Überträgt die Beschleunigungszeit von Programmplatz 1 in die numerische Variable M1
2 M1 = M_DAcl(2)      'Überträgt die Abbremszeit von Programmplatz 2 in die numerische Variable M1
3 M1 = M_NAcl         'Überträgt den Standardwert der Beschleunigungszeit von Programmplatz 1 in die numerische Variable M1
4 M1 = M_NDAcl(2)     'Überträgt den Standardwert der Abbremszeit von Programmplatz 2 in die numerische Variable M1
5 M1 = M_AclSts(3)    'Überträgt den Status der Beschleunigung/Abbremsung von Programmplatz 3 in die numerische Variable M1
```

Erläuterung

- Die Beschleunigungs-/Abremszeit wird als prozentualer Wert bezogen auf die maximale Beschleunigung/Abbremsung (Standardwert) des Roboters angegeben. Bei einem Wert von 50 % verdoppelt sich folglich die Beschleunigungs-/Abbremszeit, d. h. die Beschleunigung/Abbremsung ist geringer.
- Der Wert der Variablen M_NAcl und M_NDAcl ist 100 %.
- Die Variablen können ausschließlich gelesen werden.

7.2.14 M_BsNo

Funktion: Nummer des aktuellen Basiskoordinatensystems lesen

Die Variable enthält die Nummer des aktuellen Basiskoordinatensystems.

Eingabeformat

Bsp.: <Numerische Variable> = M_BsNo [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Beim Wert „1“ ist das Standard-Basiskoordinatensystem aktuell. Es können Konstanten, Variablen, logische/arithmetische Ausdrücke und Funktionen verwendet werden. Bei Zahlen vom Typ Real oder vom Typ Real mit doppelter Genauigkeit wird eine Nachkommastelle, die größer oder gleich 5 ist, aufgerundet, ansonsten abgerundet.

Programmbeispiel

```
1 M1 = M_BsNo                'Überträgt die Nummer des aktuellen Basis-
                             koordinatensystems in die numerische Variable M1
2 If M1 = 1 Then             'Fährt die Position P1 an, falls die Nummer des
                             Basiskoordinatensystems „1“ ist
3 Mov P1
4 Else                       'Fährt die Position P2 an, fall die Nummer des
                             Basiskoordinatensystems ungleich „1“ ist
5 Mov P2
6 EndIf
```

Erläuterung

- Die Nummer des aktuellen Basiskoordinatensystems (Parameter MEXBSNO) wird eingelesen.
- Die Koordinatensysteme sind wie folgt gekennzeichnet:
  0: Standardwert des Systems (P_NBase)
  1–8: Werkstückkoordinatensystem 1 bis 8 (Parameter WK1CORD bis WK8CORD)
  –1: Die Einstellung der Basis-Konvertierungsdaten wurde auf eine andere Weise vorgenommen (z.B. über den Base-Befehl oder den Parameter MEXBS).
- Bei einem Zugriff auf die Variable M_BrkCq wird die eingestellte Break-Bedingung gelöscht. (Der Wert wird auf „0“ gesetzt.) Möchten Sie den aktuellen Wert der Variablen weiterverwenden, weisen Sie ihn einer numerischen Variablen zu.
- Die Break-Bedingung kann auch über die Teaching Box gelöscht werden.

Steht in Beziehung zu folgenden Befehlen:

Base

Steht in Beziehung zu folgenden Parametern:

MEXBSNO, WKnCORD (n = 1 bis 8), MEXBS

7.2.15 M_BrkCq

Funktion: Break-Befehlsausführung lesen

Die Variable zeigt die Ausführung eines Break-Befehls an.
1: Break-Befehl wurde ausgeführt.
0: Break-Befehl wurde nicht ausgeführt.

Eingabeformat

Bsp.: <Numerische Variable> = M_BrkCq [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1 While M1 <> 0          'Wiederholt den Programmblock, solange M1 ungleich
                         0 ist und springt zur Zeile 4, wenn M1 gleich 0 ist
2 If M2 = 0 Then Break   'Springt zur Zeile 4, falls M2 gleich 0 ist
3 WEnd                   'Springt zurück zur While-Anweisung (Zeile 1)
4 If M_BrkCq Then Hlt    'Stoppt das Programm, wenn der Break-Befehl im
                         While-WEnd-Programmblock ausgeführt wurde
```

Erläuterung

- Mit Hilfe der Variablen kann geprüft werden, ob ein Break-Befehl ausgeführt worden ist.
- Die Variable kann ausschließlich gelesen werden.
- Nach einem einmaligen Zugriff auf die Variable M_BrkCq wird der Break-Status zurückgesetzt (d. h. die Variable wird auf „0" gesetzt). Soll der Wert weiterhin erhalten bleiben, muss er in einer numerischen Variablen gespeichert werden.
- Der Break-Status wird auch dann zurückgesetzt, wenn ein Zugriff auf die Variable zur Anzeige auf der Teaching Box o. Ä. erfolgt.

7.2.16 M_BTime

Funktion: Batterierestzeit lesen

Die Variable enthält die Restzeit der Batterie in Stunden.

Eingabeformat

Bsp.: <Numerische Variable> = M_BTime

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1  M1 = M_BTime                    'Überträgt die Restlaufzeit der Batterie in die
                                   numerische Variable M1
```

Erläuterung

- Die verbleibende Lebensdauer der Batterie wird übertragen.
- Als Standardwert für eine neue Batterie werden 14600 Stunden vorgegeben.
- Die Differenz – 14600 Stunden minus der kumulierten Einschaltzeit des Steuergerätes – wird in die Variable übertragen.
- Die Variable kann ausschließlich gelesen werden.
- Bei einem RESET wird der Wert auf 14600 zurückgesetzt.

7.2.17 M_CmpDst

Funktion: Positionsabweichung lesen

Die Variable enthält die Abweichung zwischen der Zielposition und der aktuellen Position nach Aktivierung der Achsenweichheit.

Eingabeformat

Bsp.: <Numerische Variable> = M_CmpDst [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1 Mov P1                    'Position P1 mittels Gelenk-Interpolation anfahren
2 CmpG 0.5,0.5,1.0,0.5,0.5, , ,   'Einstellung der Weichheit
3 Cmp Pos, &B00011011       'Achsenweichheit für die Richtungen X, Y und die
                            Orientierungen A und B aktivieren
4 Mvs P2                    'Position P2 mittels Linear-Interpolation anfahren
5 M_Out(10) = 1             'Setzt Ausgangsbit 10 auf „1“
6 Mvs P1                    'Position P1 mittels Linear-Interpolation anfahren
7 M1 = M_CmpDst(1)          'Setzt die Variable M1 auf den Wert der Abweichung
                            zwischen der aktuellen Position und der Zielposition
                            des Mechanismus 1
8 Cmp Off                   'Achsenweichheit deaktivieren
```

Erläuterung

- Die Variable wird bei aktivierter Achsenweichheit zur Überprüfung der Abweichung zwischen der aktuellen und der Zielposition verwendet.
- Die Variable kann ausschließlich gelesen werden.

7.2.18 M_CmpLmt

Funktion: Grenzwertüberschreitung melden

Die Variable zeigt nach Aktivierung der Achsenweichheit eine Grenzwertüberschreitung an.
1: Grenzwertüberschreitung
0: keine Grenzwertüberschreitung

Eingabeformat

Bsp.: Def Act 1, M_CmpLmt [(<Mechanismusnummer>)] = 1 GoTo *Lmt

<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  Def Act 1, M_CmpLmt(1) = 1 GoTo *Lmt   'Definiert einen Unterprogrammsprung zur Marke Lmt,
                                          wenn nach Aktivierung der Achsenweichheit ein
                                          Grenzwert überschritten wird
2                                         '
3                                         '
   :
10  Mov P1                                'Position P1 mittels Gelenk-Interpolation anfahren
11  CmpG 1,1,0,1,1,1,1,1                  'Einstellung der Weichheit
12  Cmp Pos, &B100                        'Achsenweichheit für die Richtung Z aktivieren
13  Act 1 = 1                             'Interrupt 1 freigeben
14  Mvs P2 Position                       'P2 mittels Linear-Interpolation anfahren
15                                        '
16                                        '
   :
100  *Lmt                                 'Sprungmarke Lmt festgelegt
101  Mvs P1                               'Bewegung zu Position P2 unterbrechen und
                                          Position P1 mittels Linear-Interpolation anfahren
102  Reset Err                            'Fehler zurücksetzen
103  Hlt                                  'Programmablauf unterbrechen
```

Erläuterung

- Die Variable wird verwendet, um bei einer Grenzwertüberschreitung nach Aktivierung der Achsenweichheit den Fehlerstatus durch Aufruf eines Interruptprozesses zurückzusetzen.
- Überwacht werden der Bewegungsbereich der Gelenke, die Geschwindigkeit bei aktivierter Achsenweichheit und die Abweichung zwischen Zielposition und der aktuellen Position.
- Bei ausgeschalteter Servoversorgungsspannung und bei deaktivierter Achsenweichheit ist der Variablenwert 0.
- Die Variable kann ausschließlich gelesen werden.

7.2.19 M_ColSts

Funktion: Status der Kollisionsüberwachung melden

Die Variable zeigt den Status der Kollisionsüberwachung an.
1: Kollision
0: keine Kollision

Eingabeformat

Bsp.: Def Act 1, M_ColSts [(<Mechanismusnummer>)] = 1 GoTo *LCOL,S

<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  Def Act 1,M_ColSts(1) = 1 GoTo *HOME,S   'Definiert bei einem Zusammenstoß einen
                                             Unterprogrammsprung zur Marke HOME
2  Act 1 = 1                                 'Interrupt 1 freigeben
3  ColChk On,NOERR                           'Kollisionsüberwachung ohne Fehlerausgabe
                                             aktivieren
4  Mov P1
5  Mov P2                                    'Erfolgt während der Ausführung der
                                             Programmschritte 4 bis 7 ein Zusammenstoß,
                                             wird der Interrupt-Prozess ausgeführt
6  Mov P3
7  Mov P4
8  Act 1 = 0                                 'Interrupt 1 sperren
   :
100  *HOME                                   'Interrupt-Prozess bei einem Zusammenstoß
101  ColChk Off                              'Kollisionsüberwachung deaktivieren
102  Servo On                                'Schaltet die Servospannung ein
103  PESC = P_ColDir(1)*(-2)                 'Abstand der entgegengesetzten
                                             Ausweichposition festlegen
104  PDst = P_Fbc(1) + PESC                  'Ausweichposition festlegen
105  Mvs PDst                                'Ausweichposition mittels Linear-
                                             Interpolation anfahren
106  Error 9100                              'Benutzerdefinierten, leichten Fehler
                                             ausgeben
```

Erläuterung

- Bei einem Zusammenstoß ist der Variablenwert „1“. Nach Beseitigung des Zusammenstoßes ist der Variablenwert „0“.
- Die Variable M_ColSts kann zur Definition der Interrupt-Bedingung in der Anweisung Def Act verwendet werden. Für die Kollisionsüberwachung muss dabei der NOERR-Modus festgelegt sein.
- Die Variable kann ausschließlich gelesen werden.

7.2.20 M_Cstp

Funktion: Programmstatus lesen

Die Variable zeigt den Zyklusstopp eines Programmes an.
1: Zyklusstopp (Der Zyklus wurde über die END-Taste des Steuergerätes oder ein Zyklusstoppsignal unterbrochen.)
0: kein Zyklusstopp

Eingabeformat

Bsp.: <Numerische Variable> = M_Cstp

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1 M1 = M_Cstp          'Überträgt bei einem Zyklusstopp den Wert „1“ in die
                        numerische Variable M1
```

Erläuterung

- Wird im kontinuierlichen Betrieb die END-Taste auf dem Steuergerät betätigt, erfolgt ein Zyklusstopp. Die Variable M_CSTP wird auf „1“ gesetzt.
- Die Variable kann ausschließlich gelesen werden.

7.2.21 M_Cys

Funktion: Zyklusbetrieb lesen

Die Variable zeigt den Zyklusbetrieb eines Programmes an.
1: Zyklusbetrieb (wenn das Ausführungsformat im Programmplatzparameter auf „CYC“ gesetzt ist oder der zyklische Betrieb im Befehl XRun festgelegt und der Befehl ausgeführt wurde)
0: kein Zyklusbetrieb

Eingabeformat

Bsp.: <Numerische Variable> = M_Cys

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1 M1 = M_Cys           'Überträgt im Zyklusbetrieb den Wert „1“ in die
                        numerische Variable M1
```

Erläuterung

- Das Ausführungsformat (kontinuierlich oder zyklisch) eines Programmes kann über einen Parameter oder den Startbefehl festgelegt werden. Das Ausführungsformat wird in die Variable M_Cys übertragen.
- Auch wenn das Ausführungsformat im Programmplatzparameter auf „CYC“ gesetzt ist, wird bei Festlegung des kontinuierlichen Betriebs im Startbefehl XRun die Variable auf „0“ gesetzt.
- Die Variable kann ausschließlich gelesen werden.

7.2.22 M_DIn/M_DOut

Funktion: CC-Link-Registerzugriff

Die Variable ermöglicht den Zugriff auf ein dezentrales Register in einem CC-Link-Netzwerk. Sie kann mit der Roboterserie CRnQ nicht verwendet werden.

M_DIn: Lesezugriff auf ein Eingangsregister
M_DOut: Lese- und Schreibzugriff auf ein Ausgangsregister

Eingabeformat

Bsp.: <Numerische Variable> = M_DIn (<Numerischer Ausdruck 1>)
Bsp.: <Numerische Variable> = M_DOut (<Numerischer Ausdruck 2>)

<Numerische Variable>	Legt eine numerische Variable fest, der der Registerinhalt zugewiesen wird.
<Numerischer Ausdruck 1>	Legt die Registernummer fest (≥ 6000)
<Numerischer Ausdruck 2>	Legt die Registernummer fest (≥ 6000)

Programmbeispiel

```
1 M1 = M_DIn(6000)        'Überträgt den Inhalt des CC-Link-Eingangsregisters
                          in die numerische Variable M1 (wenn die CC-Link-
                          Stationsnummer gleich 1 ist)
2 M1 = M_DOut(6000)       'Überträgt den Inhalt des CC-Link-Ausgangsregisters
                          in die numerische Variable M1
3 M_DOut(6000) = 100      'Schreibt den Wert „100“ in das CC-Link-Ausgangs-
                          register
```

Erläuterung

- Detaillierte Hinweise finden Sie im Handbuch des CC-Link-Schnittstellenmoduls.
- Signalnummern ab 6000 werden für den CC-Link-Zugriff verwendet.
- Die Variable M_DIn kann ausschließlich gelesen werden.

7.2.23 M_Err/M_ErrLvl/M_Errno

Funktion: Fehlerdaten lesen

Die Variable ermöglicht den Zugriff auf die Daten eines vom Roboter generierten Fehlers.

M_Err: Fehlerstatus
M_ErrLvl: Fehlerklassen
M_Errno: Fehlernummer

Eingabeformat

Bsp.: <Numerische Variable> = M_Err

Bsp.: <Numerische Variable> = M_ErrLvl

Bsp.: <Numerische Variable> = M_Errno

<Numerische Variable> Legt eine numerische Variable fest
Bedeutung:
M_Err: 0/1 = kein Fehler/Fehler
M_ErrLvl: 0/1/2/3/4/5/6 = kein Fehler/Warnung/leichter Fehler/schwerer Fehler/Warnung 1/leichter Fehler 1/schwerer Fehler 1 (die Bedeutung der Fehler entnehmen Sie Tab. 7-1.)
M_ErrNo: enthält die Nummer des generierten Fehlers

Programmbeispiel

```
1 *REP: IF M_ERR = 0 Then GoTo *REP    'Wartestatus, bis ein Fehler auftritt
2 M2 = M_ErrLvl                        'Überträgt die Fehlerklasse in M2
3 M3 = M_Errno                         'Überträgt die Fehlernummer in M3
```

Erläuterung

- In der Regel werden Programme unterbrochen, wenn ein Fehler (nicht bei Warnmeldungen) auftritt. Mit Hilfe der Variablen kann der Fehlerstatus von Programmen angezeigt werden, deren Startbedingung in den Programmplatzparametern auf „ALWAYS“ gesetzt ist. Ein Programm mit der Startbedingung „ALWAYS“ wird auch dann nicht unterbrochen, wenn in einem anderen Programm ein Fehler auftritt.
- Treten zwei oder mehr Fehler auf, wird die Information der höchsten Fehlerklasse übertragen.

- Je nach Fehlerklasse (M_ErrLvl) reagiert das System wie folgt:

Fehlerklasse	Bezeichnung	Beschreibung	Fehler zurücksetzen
0	Kein Fehler	Es ist kein Fehler aufgetreten.	—
1	Warnung	Das Programm wird fortgesetzt.	[RESET]-Taste
2	Leichter Fehler	Das Programm wird unterbrochen.	
3	Schwerer Fehler	Das Programm wird unterbrochen und die Servoversorgung abgeschaltet.	
4	Warnung 1	Das Programm wird fortgesetzt.	Spannungsversorgung aus- und wieder einschalten
5 ①	Leichter Fehler 1	Das Programm wird unterbrochen.	
6 ①	Schwerer Fehler 1	Das Programm wird unterbrochen und die Servoversorgung abgeschaltet.	

Tab. 7-1: *Fehlerklassen*

① Die Fehlerklassen 5 und 6 stehen ab folgender Software-Version zur Verfügung:
SD-Serie: S1d oder später
SQ-Serie: R1d oder später

Steht in Beziehung zu folgenden Befehlen:

Error, Reset Err

7.2.24 M_Exp

Funktion: Basis des natürlichen Logarithmus lesen

Die Variable enthält den Wert der Basis des natürlichen Logarithmus (2,718281828459045).

Eingabeformat

Bsp.: <Numerische Variable> = M_Exp

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1  M1 = M_Exp                    'Überträgt den Wert der Basis des natürlichen
                                 Logarithmus (2,71828) in die numerische
                                 Variable M1
```

Erläuterung

- Die Variable dient zur Berechnung von Exponential- und Logarithmusfunktionen.
- Die Variable kann ausschließlich gelesen werden.

7.2.25 M_Fbd

Funktion: Differenz zwischen Soll- und Istposition lesen

Die Variable enthält die Differenz zwischen der durch den Befehlswert vorgegebenen Sollpostion und der durch die Encoderimpulse gemeldeten Istposition.

Eingabeformat

Bsp.: <Numerische Variable> = M_Fbd[(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  Def Act 1,M_Fbd > 10 GoTo *SUB1,S   'Definiert einen Unterprogrammsprung zur Marke
                                       SUB1, wenn die Abweichung zwischen der Soll- und
                                       der Istpostion größer gleich 10 mm ist
2  Act 1 = 1                           'Interrupt 1 freigeben
3  Torq 3,10                           'Legt die Drehmomentgrenze für Achse 3 auf 10 %
                                       des Maximalwerts fest
4  Mvs P1                              'Position P1 mittels Linear-Interpolation anfahren
5  Mov P2                              'Position P2 mittels Gelenk-Interpolation anfahren
   :
10 *SUB1                               'Sprungmarke SUB1 festgelegt
11 Mov P_Fbc                           'Soll- und Istposition abgleichen
12 M_Out(10) = 1                       'Setzt das allgemeine Ausgangssignal 10 auf „1“
13 Hlt                                 'Programm stoppen, wenn die Abweichung zwischen
                                       Soll und Istposition größer 10 mm ist
```

Erläuterung

- Die Variable enthält die Differenz zwischen der durch den Befehlswert vorgegebenen Position und der durch den Motor rückgemeldeten Position. Bei Verwendung des Torq-Befehls sollte die Variable zur Vermeidung von zu großen Positionsabweichungen (Fehler H0960, H0970 usw.) zusammen mit dem Def-Act-Befehl verwendet werden.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Befehlen:

Torq

Steht in Beziehung zu folgenden Variablen:

P_Fbc

7.2.26 M_G

Funktion: Erdbeschleunigung lesen

Die Variable enthält den Wert der Erdbeschleunigung (9,80665).

Eingabeformat

Bsp.: <Numerische Variable> = M_G

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1  M1 = M_G          'Überträgt den Wert der Erdbeschleunigung (9,80665)
                     in die numerische Variable M1
```

Erläuterung

- Die Variable dient zur Berechnung von Funktionen mit Hilfe der Erdbeschleunigung.
- Die Variable kann ausschließlich gelesen werden.

7.2.27 M_HndCq

Funktion: Handgreiferzustand lesen

Die Variable enthält Daten über den Handgreiferzustand.

Eingabeformat

Bsp.: <Numerische Variable> = M_HndCq (<Numerischer Ausdruck>)

<Numerische Variable> Legt eine numerische Variable fest

<Numerischer Ausdruck> Legt die Handeingangsnummer fest
$1 \leq$ Handeingangsnummer ≤ 8
(entspricht den Eingangssignalen 900 bis 907)

Programmbeispiel

```
1  M1 = M_HndCq      'Überträgt Handgreiferzustand der Hand 1 in die
                     numerische Variable M1
```

Erläuterung

- Die Variable enthält ein Bit des Handgreiferzustands (z. B. ein Sensorsignal).
- Die Angabe M_HndCq(1) bezeichnet im Grundzustand die Eingangssignalnummer 900, d. h. sie entspricht der Angabe M_In(900), und kann umparametriert werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.28 M_In/M_Inb/M_In8/M_Inw/M_In16

Funktion: Eingangssignal lesen

Die Variablen enthalten die Werte verschiedener Eingangssignale.

M_In:	Lesezugriff auf ein Eingangssignalbit
M_Inb oder M_In8:	Lesezugriff auf ein Eingangssignalbyte (8 Bit)
M_Inw oder M_In16:	Lesezugriff auf ein Eingangssignalwort (16 Bit)

Eingabeformat

Bsp.: <Numerische Variable> = M_In (<Numerischer Ausdruck>)
Bsp.: <Numerische Variable> = M_Inb (<Numerischer Ausdruck>) oder M_In8 (<Numerischer Ausdruck>)
Bsp.: <Numerische Variable> = M_Inw (<Numerischer Ausdruck>) oder M_In16 (<Numerischer Ausdruck>)

<Numerische Variable> Legt eine numerische Variable fest
Die entsprechende Zuordnung zeigt Tab. 7-2.

<Numerischer Ausdruck> Legt die Eingangssignalnummer fest
Die entsprechende Zuordnung zeigt Tab. 7-3.
Serie CRnD-700:
- 0 bis 255: Standardeingänge
- 716 bis 731: Reserviert
- 900 bis 907: Handeingang
- 2000 bis 5071: Eingangssignale in einem PROFIBUS-Netzwerk
- 6000 bis 8047: dezentrales Eingangsregister in einem CC-Link-Netzwerk

Serie CRnQ-700:
- 716 bis 731: Reserviert
- 900 bis 907: Handeingang
- 10000 bis 18191: Übergreifende Mehrfach-CPU-Operanden

Programmbeispiel

```
1 M1% = M_In(0)          'Überträgt den Wert des Eingangssignals 0 (0 oder 1) in die numerische Variable M1
2 M2% = M_Inb(0)         'Überträgt den 8-Bit-Wert des Eingangssignals, beginnend mit Bit 0, in die numerische Variable M2
3 M3% = M_Inb(3) AND &H7 'Überträgt den 3-Bit-Wert des Eingangssignals, beginnend mit Bit 3, in die numerische Variable M3
4 M4% = M_Inw(5)         'Überträgt den 16-Bit-Wert des Eingangssignals, beginnend mit Bit 5, in die numerische Variable M4
```

Erläuterung

- Die Variablen enthalten die Zustände verschiedener Eingangssignale.
- M_Inb/M_In8 und M_Inw/M_In16 enthalten 8-Bit- oder 16-Bit-Informationen, beginnend mit dem festgelegten Bit.
- Auch wenn Signalnummern bis 32767 verwendet werden können, muss über die Nummern ein sinnvoller Zugriff auf die vorhandenen Eingänge erfolgen. Bei Angabe von Signalnummern, denen keine Hardware zugeordnet ist, wird ein undefinierter Zustand gesetzt.
- Die Variablen können ausschließlich gelesen werden.

ACHTUNG:

Unterbrechen Sie immer die Signale, um eine Synchronisation auszulösen. Andernfalls können durch falsche oder unvollständig übertragene Signale Fehlfunktionen auftreten.

Bit-Anzahl	Numerische Variablen				Sonstige Variablen		
	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit Bsp.: M1!	Real-Wert mit doppelter Genauigkeit Bsp.: M1#	Position ① Bsp.: P1.X	Gelenk ① Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_In	●	●	●	●	●	●	○
M_Inb/M_In8	●	●	●	●	●	●	○
M_Inw/M_In16	●	●	●	●	●	●	○

Tab. 7-2: *Numerische Variable*

●: Verfügbar

○: Nicht verfügbar

② Wenn die Variable den Wert eines Winkels hat, so ist in diesem Fall die Einheit „Radiant". (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Zur Anzeige wird der Wert umgewandelt und in der Einheit „Grad" dargestellt.
Beispiel:
Ist in P1.A = M_In(8) die Eingangssignalnummer 8 eingeschaltet, wird für P1.A der Wert „57.296" angezeigt. Daraus ist der EIN/AUS-Status nicht klar erkennbar. Da die Einheit der Komponenten X, Y und Z der Positionsvariable „mm" ist, existiert dieser Zustand nicht.

Bit-Anzahl	Konstanten			Numerische Variablen				Sonstige Variablen		
	Numerischer Wert ① Bsp.: 12	Binäre Zahl Bsp.: &B1100	Hexadezimale Zahl Bsp.: &HC	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit Bsp.: M1!	Real-Wert mit doppelter Genauigkeit Bsp.: M1#	Position ①② Bsp.: P1.X	Gelenk ①② Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_In	●	●	●	●	●	●	●	●	●	○
M_Inb/M_In8	●	●	●	●	●	●	●	●	●	○
M_Inw/M_In16	●	●	●	●	●	●	●	●	●	○

Tab. 7-3: *Numerischer Ausdruck*

●: Verfügbar

○: Nicht verfügbar

① Real-Wert wird abgerundet

② Wenn die Variable den Wert eines Winkels hat, wird diese mit der Einheit „Radiant" verarbeitet. (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Daher ist die Zuordnung der Signalnummer sehr schwierig. Zur Anzeige wird der Wert in „Grad" umgewandelt und es wird der gleiche Wert, wie der Einstellwert dargestellt.
Beispiel:
Es wird der Wert „0" verarbeitet, obwohl für P1.A zur Festlegung der Eingangssignalnummer der Wert „8" eingestellt wurde (z. B. über Eingabe mit der Teaching Box-Tastatur usw.). Der Wert „8" (Grad) wird in „0,14" (Radiant) umgewandelt, was nach Abrundung den Wert „0" ergibt. Da die Einheit der Komponenten X, Y und Z der Positionsvariable „mm" ist, existiert dieser Zustand nicht.

Steht in Beziehung zu folgenden Variablen:

M_Out32

Steht in Beziehung zu folgenden Befehlen:

Def IO

7.2.29 M_In32

Funktion: Eingangssignal lesen

Die Variable enthält den Wert des Doppelworts (32 Bit) eines Eingangssignals.

Eingabeformat

Bsp.: <Numerische Variable> = M_In32 (<Numerischer Ausdruck>)

<Numerische Variable> Legt eine numerische Variable fest
Die entsprechende Zuordnung zeigt Tab. 7-4.

<Numerischer Ausdruck> Legt die Eingangssignalnummer fest
Die entsprechende Zuordnung zeigt Tab. 7-5.
Serie CRnD-700:
- 0 bis 255: Standardeingänge
- 716 bis 731: Reserviert
- 900 bis 907: Handeingang
- 2000 bis 5071: Eingangssignale in einem PROFIBUS-Netzwerk
- 6000 bis 8047: dezentrales Eingangsregister in einem CC-Link-Netzwerk

Serie CRnQ-700:
- 716 bis 731: Reserviert
- 900 bis 907: Handeingang
- 10000 bis 18191: Übergreifende Mehrfach-CPU-Operanden

Programmbeispiel

```
1 *ack_wait
2 If M_In(7) = 0 Then *ack_wait      'Warten, bis Eingangssignal 7 einschaltet (Unterbrechung).
3 M1& = M_In32(10000)                'Speichert den 32-Bit-Datenwert der Eingangssignalnr. 10000 in die Variable M1 mit Doppelwort-Genauigkeit
4 P1.Y=M_In32(10100)/1000            'Eingabe des Ergebniswerts der Division des 32-Bit-Datenwerts der Eingangssignalnr. 10100 mit dem Wert 1000 in die Y-Komponente der Positionsvariable P1 (Dieses Beispiel setzt voraus, dass das externe Signal ursprünglich ein Real-Wert war, danach mit 1000 multipliziert und in einen Integer-Wert umgewandelt wurde.)
```

Erläuterung

- Die Variable enthält die Eingangssignaldaten als numerischen Wert mit 32-Bit-Länge.
- Auch wenn Signalnummern bis 32767 verwendet werden können, muss über die Nummern ein sinnvoller Zugriff auf die vorhandenen Eingänge erfolgen. Bei Angabe von Signalnummern, denen keine Hardware zugeordnet ist, wird ein undefinierter Zustand gesetzt.
- Als <Numerische Variable> kann ein Integer-Wert mit Doppelwort-Genauigkeit oder ein Real-Wert eingesetzt werden.
- Die Variablen können ausschließlich gelesen werden.

ACHTUNG:

Unterbrechen Sie immer die Signale, um eine Synchronisation auszulösen. Andernfalls können durch falsche oder unvollständig übertragene Signale Fehlfunktionen auftreten.

Bit-Anzahl	Numerische Variablen				Sonstige Variablen		
	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit Bsp.: M1!	Real-Wert mit doppelter Genauigkeit Bsp.: M1#	Position ① Bsp.: P1.X	Gelenk ① Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_In32	○	●	●	●	●	●	○

Tab. 7-4: *Numerische Variable*

●: Verfügbar

○: Nicht verfügbar

① Wenn die Variable den Wert eines Winkels hat, so ist in diesem Fall die Einheit „Radiant". (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Zur Anzeige wird der Wert umgewandelt und in der Einheit „Grad" dargestellt.
Beispiel:
Wenn der Eingangsdatenwert in P1.A = M_In32(16) die dezimale Zahl „5" ist, wird für P1.A der Wert „286.479" (Grad) angezeigt.

Bit-Anzahl	Konstanten			Numerische Variablen				Sonstige Variablen		
	Numerischer Wert ① Bsp.: 12	Binäre Zahl Bsp.: &B1100	Hexadezimale Zahl Bsp.: &HC	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit Bsp.: M1!	Real-Wert mit doppelter Genauigkeit Bsp.: M1#	Position ①② Bsp.: P1.X	Gelenk ①② Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_In32	●	●	●	●	●	●	●	●	●	○

Tab. 7-5: *Numerischer Ausdruck*

●: Verfügbar

○: Nicht verfügbar

① Real-Wert wird abgerundet

② Wenn die Variable den Wert eines Winkels hat, wird diese mit der Einheit „Radiant" verarbeitet. (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Daher ist die Zuordnung der Signalnummer sehr schwierig. Zur Anzeige wird der Wert in „Grad" umgewandelt und es wird der gleiche Wert, wie der Einstellwert dargestellt.
Beispiel:
Es wird der Wert „0" verarbeitet, obwohl für P1.A zur Festlegung der Eingangssignalnummer der Wert „8" eingestellt wurde (z. B. über Eingabe mit der Tastatur der Teaching Box usw.). Der Wert „8" (Grad) wird in „0,14" (Radiant) umgewandelt, was nach Abrundung den Wert „0" ergibt. Da die Einheit der Komponenten X, Y und Z der Positionsvariable „mm" ist, existiert dieser Zustand nicht.

Steht in Beziehung zu folgenden Variablen:

M_In/M_Inb/M_In8/M_Inw/M_In16

Steht in Beziehung zu folgenden Befehlen:

Def IO

7.2.30 M_JOvrd/M_NJOvrd/M_OPOvrd/M_Ovrd/M_NOvrd

Funktion: Übersteuerungswerte lesen

Die Variablen enthalten Übersteuerungswerte.

M_JOvrd:	Wert der über die JOvrd-Anweisung festgelegten Übersteuerung für die Gelenk-Interpolation
M_NJovrd:	Standard-Übersteuerungswert für die Gelenk-Interpolation (100 %)
M_OPOvrd:	über das Steuergerät eingestellter Übersteuerungswert
M_Ovrd:	Wert der über die Ovrd-Anweisung festgelegten aktuellen Übersteuerung
M_NOvrd:	Standard-Übersteuerungswert (100 %)

Eingabeformat

Bsp.: <Numerische Variable> = M_JOvrd [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_NJOvrd [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_OPOvrd

Bsp.: <Numerische Variable> = M_Ovrd [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_NOvrd [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz (Slot) gesetzt.

Programmbeispiel

```
1 M1 = M_Ovrd          'Überträgt den über die OVRD-Anweisung festgelegten aktuellen Übersteuerungswert in die numerische Variable M1
2 M2 = M_NOvrd         'Überträgt den Standard-Übersteuerungswert von 100 % in die numerische Variable M2
3 M3 = M_JOvrd         'Überträgt den Übersteuerungswert für die Gelenk-Interpolation in die numerische Variable M3
4 M4 = M_NJovrd        'Überträgt den Standard-Übersteuerungswert für die Gelenk-Interpolation in die numerische Variable M4
5 M5 = M_OPOvrd        'Überträgt den über das Steuergerät festgelegten Übersteuerungswert in die numerische Variable M5
6 M6 = M_Ovrd(2)       'Überträgt den aktuellen Übersteuerungswert von Programmplatz 2 in die numerische Variable M6
```

Erläuterung

- Fehlt die Angabe der Programmplatznummer, wird der Wert des aktuell ausgewählten Programmplatzes übertragen.
- Die Variablen können ausschließlich gelesen werden.

7.2.31 M_LdFact

Funktion: Lastverhältnis lesen

Die Variable enthält das Lastverhältnis der einzelnen Gelenkachsen.

Eingabeformat

Bsp.: <Numerische Variable> = M_LdFact (<Achsennummer>)

<Numerische Variable>	Legt eine numerische Variable im Bereich von 0 bis 100 % fest
<Achsennummer>	Legt die Achsennummer fest 1 ≤ Achsennummer ≤ 8

Programmbeispiel

```
1  Accel 100,100                'Beschleunigung/Abbremsung für Standardlast
                                einstellen
2  *Label                       'Sprungmarke Label festgelegt
3  Mov P1                       'Position P1 mittels Gelenk-Interpolation anfahren
4  Mov P2                       'Position P2 mittels Gelenk-Interpolation anfahren
5  If M_LdFact(2) > 90 Then     'Bei Überschreitung des Lastverhältnisses von 90 %,
                                gehe zur folgenden Zeile
6     Accel 50,50               'Beschleunigung/Abbremsung auf 50 % reduzieren
7     M_SetAdl(2) = 50          'Zusätzlich wird die Beschleunigung/Abbremsung der
                                Achse J2 um 50 % auf effektiv 25 % verkleinert
                                (d. h.: 50 % × 50 % = 25 %)
8  Else
9     Accel 100,100             'Beschleunigung/Abbremsung wieder auf Standard-
                                last einstellen
10 Endlf
11 GoTo *Label                  'Sprung zur Marke Label
```

Erläuterung

- Die Variable ermöglicht einen Zugriff auf das Lastverhältnis der einzelnen Achsen.
- Das Lastverhältnis wird aus den einzelnen Servomotorströmen und der Zeit, in der die Ströme fließen, ermittelt.
- Das Lastverhältnis steigt, wenn der Roboter über einen längeren Zeitraum mit einer schweren Last in einer belastungsintensiven Stellung betrieben wird.
- Erreicht das Lastverhältnis 100 % erfolgt die Meldung eines Überlastfehlers. Im Programmbeispiel erfolgt bei einem Lastverhältnis von 90 % eine Verringerung der Beschleunigung/Abbremsung auf 25 %.
- Folgende Maßnahmen dienen der Verkleinerung des Lastverhältnisses:
  - Beschleunigungs-/Bremszeit vergrößern
  - weniger belastungsintensive Stellungen für die Verfahrbewegungen wählen
  - Servoversorgungsspannung abschalten
- Bei der Software-Version N8 (Serie SQ), bzw. P8 (Serie SD) ist der Mechanismus standardmäßig mit der Nummer „1“ definiert. Wenn nach der Ausführung des RelM-Befehls dem Mechanismus die Nummer „1“ zugeordnet wurde oder wenn die Programmplatznummer (Slot) ungleich „1“ ist, dann ist die Ausführung des GetM-Befehls nicht notwendig. Ist die Mechanismusnummer nicht „1“, muss zuvor der GetM-Befehl ausgeführt werden.

Steht in Beziehung zu folgenden Befehlen:

Accel, Ovrd

Steht in Beziehung zu folgenden Variablen:

M_SetAdl

7.2.32 M_Line

Funktion: Zeilennummer lesen

Die Variable enthält die Nummer der Zeile, die gerade ausgeführt wird.

Eingabeformat

Bsp.: <Numerische Variable> = M_LINE [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1  M1 = M_Line(2)        'Überträgt die Nummer der Zeile, die gerade im
                         Programmplatz 2 ausgeführt wird, in die numerische
                         Variable M1
```

Erläuterung

- Die Variable kann im Multitasking-Betrieb zur Überwachung der aktuell ausgeführten Zeile durch andere Programmplätze verwendet werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.33 M_Mode

Funktion: Betriebsart lesen

Die Variable enthält die über den MODE-Umschalter eingestellte Betriebsart.
1: MANUAL
2: AUTOMATIC

Eingabeformat

Bsp.: <Numerische Variable> = M_Mode

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1 M1 = M_Mode          'Überträgt die über den MODE-Schalter eingestellte
                       Betriebsart in die numerische Variable M1
```

Erläuterung

- Die Variable kann im Multitasking-Betrieb zur Überwachung kontinuierlich ausgeführter Programme (Startbedingung: ALWAYS) verwendet werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.34 M_On/M_Off

Funktion: Signalzustand schalten

Die Variable überträgt den Wert „1“ (M_On) oder „0“ (M_Off).

Eingabeformat

Bsp.: <Numerische Variable> = M_On
Bsp.: <Numerische Variable> = M_Off

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1 M1 = M_On            'Setzt M1 auf „1“
2 M2 = M_Off           'Setzt M2 auf „0“
```

Erläuterung

- Die Variable überträgt den Signalzustand „1“ oder „0“.
- Die Variable kann ausschließlich gelesen werden.

7.2.35 M_Open

Funktion: Dateistatus lesen

Die Variable zeigt an, ob eine Datei geöffnet ist oder nicht.
Der Status des RS232C-Kanals auf der Rechnerseite wird übertragen.

Eingabeformat

Bsp.: <Numerische Variable> = M_Open [<Dateinummer>]

<Numerische Variable>	Legt eine numerische Variable fest
<Dateinummer>	Gibt die Nummer eines mit dem Open-Befehl geöffneten Kommunikationskanals als Konstante im Bereich zwischen 1 und 8 an (Standardwert: 1) Bei Angabe einer Dateinummer ≥ 9 erfolgt eine Fehlermeldung.

Programmbeispiel

```
1  Open "COM2:" AS #1                          'Öffnet den Kommunikationskanal COM 2 als
                                               Datei Nr. 1
2  *LBL:If M_Open(1) <> 1 Then GoTo *LBL       'Wartestatus, bis die Datei Nummer 1 geöffnet wird

<Bei Verwendung der Ethernet-Schnittstelle>
1  ' Client-Programm ----------
2  M1 = 0
3  M_Timer(1) = 0                              'Zähler zurücksetzen
4  *O1                                         'Sprungmarke O1 festgelegt
5  Open "COM2:" As #1                          'Öffnet den Kommunikationskanal COM 2 als
                                               Datei Nr. 1
6  If M_Timer(1) > 10000.0 Then *E1            'Nach Ablauf von 10 Sek. Sprung zur Marke E1
7  If M_Open(1) <> 1 Then Goto *O1             'Wartestatus, bis die Datei Nummer 1 geöffnet wird
8  Def Act 1,M_Open(1) = 0 GoSub *E2           'Definiert einen Unterprogrammsprung zur Marke
                                               E2, wenn der Server außer Betrieb ist
9  Act 1 = 1                                   'Interrupt 1 freigeben
10 *M1
11 M1 = M1 + 1
12 If M1 < 10 Then C1$ = "MELFA" Else C1$="END"'   'Nach neunmaliger Übertragung der Zeichen-
                                                    kette „MELFA“ wird „END“ übertragen.
13 Print #1,C1$                                'Sendet die Zeichenkette C1$
14 Input #1,C2$                                'Liest die Zeichenkette C2$.
15 If C1$="END" Then *C1                       'Springt zur Marke C1 nach Senden von „END“
16 GoTo *M1                                    'Schleifensprung zu Marke M1
17 *C1
18 Close #1                                    'Schließt den Kommunikationskanal
19 Hlt                                         'Programmablauf unterbrechen
20 End                                         'Programmende
21 *E1
22 Error 9100                                  'Fehler 9100 ausgeben, wenn Server außer Betrieb
23 Close #1
24 Hlt
25 End
26 *E2                                         'Sprungmarke E2 festgelegt
27 Error 9101                                  'Fehler 9101ausgeben, wenn Server während des
                                               Ablaufs außer Betrieb geht
28 Close #1
29 Hlt
30 End
```

Erläuterung

- Die Variable kann ausschließlich gelesen werden.
- Der Variablenwert ist vom Typ der mit dem Open-Befehl geöffneten Datei abhängig:

Dateityp	Beschreibung	Variablenwert
Datei	Die Variable zeigt an, ob eine Datei geöffnet ist oder nicht. Nach Ausführung des Befehls Open ist der Variablenwert solange „1", bis einer der Befehle Close oder End ausgeführt oder das Programmende erreicht wird.	1: bereits geöffnet –1: nicht definierte Dateinummer (nicht geöffnet)
Kommunikationskanal RS232C	Die Variable zeigt den Status der RS232C-Kommunikationsleitung an. Es wird der Zustand des CTS-Signals angezeigt. (Die Funktion kann nur nach Freigabe des RTS-Signals am rechnerseitigen Leitungsende und bei Verwendung des Mitsubishi-Kabels RV-CAB genutzt werden.)	1: bereits verbunden (CTS-Signal EIN) 0: nicht verbunden (CTS-Signal AUS) –1: nicht definierte Dateinummer (nicht geöffnet)

Tab. 7-6: *Variablenwert und Bedeutung*

Steht in Beziehung zu folgenden Befehlen:

Open

Steht in Beziehung zu folgenden Parametern:

COMDEV, CPRE**, NETMODE

7.2.36 M_Out/M_Outb/M_Out8/M_Outw/M_Out16

Funktion: Ausgangssignal lesen/schreiben

Die Variablen ermöglichen den Schreib- und Lesezugriff auf externe Ausgangssignale.

M_Out:	Schreib-/Lesezugriff auf ein Ausgangssignalbit
M_Outb oder M_Out8:	Schreib-/Lesezugriff auf ein Ausgangssignalbyte (8 Bit)
M_Outw oder M_Out16:	Schreib-/Lesezugriff auf ein Ausgangssignalwort (16 Bit)

Eingabeformat

Bsp.: M_Out (<Numerischer Wert 1>) = <Numerischer Wert 2>

Bsp.: M_Outb (<Numerischer Wert 1>) oder M_Out8 (<Numerischer Wert 1>) = <Numerischer Wert 3>

Bsp.: M_Outw (<Numerischer Wert 1>) oder M_Out16 (<Numerischer Wert 1>) = <Numerischer Wert 4>

Bsp.: M_Out (<Numerischer Wert 1>) = <Numerischer Wert 2> Dly <Zeit>

Bsp.: <Numerische Variable> = M_Out (<Numerischer Wert 1>)

<Numerischer Wert 1> Legt die Ausgangssignalnummer fest
Die entsprechende Zuordnung zeigt Tab. 7-7.
Serie CRnD-700:
- 0 bis 255: Standardausgänge
- 716 bis 723: Reserviert
- 900 bis 907: Handausgang
- 2000 bis 5071: Eingangssignale in einem PROFIBUS-Netzwerk
- 6000 bis 8047: dezentrales Eingangsregister in einem CC-Link-Netzwerk

Serie CRnQ-700:
- 716 bis 723: Reserviert
- 900 bis 907: Handausgang
- 10000 bis 18191: Übergreifende Mehrfach-CPU-Operanden

<Numerische Variable> Legt eine numerische Variable fest

<Numerischer Wert 2>
<Numerischer Wert 3>
<Numerischer Wert 4> Legt den Wert fest, der von einer numerischen Variablen, einer Konstanten oder einem numerisch-arithmetischen Ausdruck ausgegeben werden soll
Die entsprechende Zuordnung zeigt Tab. 7-8.
Numerische Bereiche:
- <Numerischer Wert 2>: 0 oder 1 (&H0 oder &H1)
- <Numerischer Wert 3>: –128 bis +127 (&H80 bis &H7F)
- <Numerischer Wert 4>: –32768 bis +32767 (&H8000 bis &H7FFF)

<Zeit> Legt die Impulsdauer in Sekunden als Konstante oder als numerischen Ausdruck fest

Programmbeispiel

```
1 M_Out(2) = 1                  'Einschalten des Ausgangssignals 2 (1 Bit)
2 M_Outb(2) = &HFF              'Einschalten von 8 Bits, beginnend mit dem Bit 2
3 M_Outw(2) = &HFFFF            'Einschalten von 16 Bits, beginnend mit dem Bit 2
4 M4 = M_Outb(2) AND &H0F       'Überträgt den 4-Bit-Wert des Ausgangssignals,
                                 beginnend mit dem Bit 2, in die numerische
                                 Variable M4
```

Erläuterung

- Die Variablen ermöglichen den Schreib- und Lesezugriff auf externe Ausgangssignale.
- Signalnummern ab 6000 werden für den CC-Link-Zugriff verwendet (optional).
- Im Abschn. 6.3.29 „Dly (Delay)" wird die Festlegung der Impulsdauer für die Impulsausgabe beschrieben.
- Die Taktzeit zur Aktualisierung der externen Ausgangssignale kann durch die Einstellung des Parameters SYNCIO auf „High-Speed" verkürzt werden. Die entsprechende Einstellung ist in Abschnitt 9.3 beschrieben.

ACHTUNG:

Unterbrechen Sie immer die Signale, um eine Synchronisation auszulösen. Andernfalls können durch falsche oder unvollständig übertragene Signale Fehlfunktionen auftreten.

Bit-Anzahl	Konstanten			Numerische Variablen				Sonstige Variablen		
	Numerischer Wert ①	Binäre Zahl	Hexadezimale Zahl	Integer	Integer-Wert mit Doppelwort-Genauigkeit	Real-Wert mit einfacher Genauigkeit	Real-Wert mit doppelter Genauigkeit	Position ①②	Gelenk ①②	Zeichenkette
	Bsp.: 12	Bsp.: &B1100	Bsp.: &HC	Bsp.: M1%	Bsp.: M1&	Bsp.: M1!	Bsp.: M1#	Bsp.: P1.X	Bsp.: J1.J1	Bsp.: C1$
M_Out	●	●	●	●	●	●	●	●	○	○
M_Outb/ M_Out8	●	●	●	●	●	●	●	●	○	○
M_Outw/ M_Out16	●	●	●	●	●	●	●	●	○	○

Tab. 7-7: *Numerischer Wert 1*

●: Verfügbar

○: Nicht verfügbar

① Real-Wert wird abgerundet

② Wenn die Variable den Wert eines Winkels hat, wird diese mit der Einheit „Radiant" verarbeitet. (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Daher ist die Zuordnung der Signalnummer sehr schwierig. Zur Anzeige wird der Wert in „Grad" umgewandelt und es wird der gleiche Wert, wie der Einstellwert dargestellt.
Beispiel:
Es wird der Wert „0" verarbeitet, obwohl für P1.A zur Festlegung der Eingangssignalnummer der Wert „8" eingestellt wurde (z. B. über Eingabe mit der Tastatur der Teaching Box usw.). Der Wert „8" (Grad) wird in „0,14" (Radiant) umgewandelt, was nach Abrundung den Wert „0" ergibt. Da die Einheit der Komponenten X, Y und Z der Positionsvariable „mm" ist, existiert dieser Zustand nicht.

Bit-Anzahl	Konstanten			Numerische Variablen				Sonstige Variablen		
	Numerischer Wert Bsp.: 12	Binäre Zahl Bsp.: &B1100	Hexadezimale Zahl Bsp.: &HC	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit Bsp.: M1!	Real-Wert mit doppelter Genauigkeit Bsp.: M1#	Position Bsp.: P1.X	Gelenk Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_Out	●	●	●	●	○	○	○	○	○	○
M_Outb/ M_Out8	●	●	●	●	○	○	○	○	○	○
M_Outw/ M_Out16	●	●	●	●	○	○	○	○	○	○

Tab. 7-8: *Numerischer Wert 2, Numerischer Wert 3, Numerischer Wert 4*

●: Verfügbar

○: Nicht verfügbar

Bit-Anzahl	Numerische Variablen				Sonstige Variablen		
	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit Bsp.: M1!	Real-Wert mit doppelter Genauigkeit Bsp.: M1#	Position ① Bsp.: P1.X	Gelenk ① Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_Out	●	●	●	●	●	●	○
M_Outb/ M_Out8	●	●	●	●	●	●	○
M_Outw/ M_Out16	●	●	●	●	●	●	○

Tab. 7-9: *Numerischer Wert*

●: Verfügbar

○: Nicht verfügbar

① Wenn die Variable den Wert eines Winkels hat, wird diese mit der Einheit „Radiant“ verarbeitet. (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Daher ist die Zuordnung der Signalnummer sehr schwierig. Zur Anzeige wird der Wert in „Grad“ umgewandelt und es wird der gleiche Wert, wie der Einstellwert dargestellt.

Steht in Beziehung zu folgenden Variablen:

M_Out32

Steht in Beziehung zu folgenden Befehlen:

Def IO

Steht in Beziehung zu folgenden Parametern:

SYNCIO

7.2.37 M_Out32

Funktion: Ausgangssignal lesen/schreiben

Die Variable ermöglicht den Schreib- und Lesezugriff auf das Doppelwort (32 Bit) eines externen Ausgangssignals.

Eingabeformat

Bsp.: M_Out32 (<Numerischer Wert 1>) = <Numerischer Wert>

Bsp.: <Numerische Variable> = M_Out32 (<Numerischer Wert 1>)

<Numerischer Wert 1> Legt die Ausgangssignalnummer fest
Die entsprechende Zuordnung zeigt Tab. 7-10.
Serie CRnD-700:
0 bis 255: Standardausgänge
716 bis 723: Reserviert
900 bis 907: Handausgang
2000 bis 5071: Eingangssignale in einem PROFIBUS-Netzwerk
6000 bis 8047: dezentrales Eingangsregister in einem CC-Link-Netzwerk
Serie CRnQ-700:
716 bis 723: Reserviert
900 bis 907: Handausgang
10000 bis 18191: Übergreifende Mehrfach-CPU-Operanden

<Numerischer Wert> Legt den Wert fest, der von einer numerischen Variablen, einer Konstanten oder einem numerisch-arithmetischen Ausdruck ausgegeben werden soll
Die entsprechende Zuordnung zeigt Tab. 7-8.
Numerischer Bereich:
–2147483648 bis +2147483647 (&H80000000 bis &H7FFFFFFF)

<Numerische Variable> Legt eine numerische Variable fest
Die entsprechende Zuordnung zeigt Tab. 7-10.

Programmbeispiel

```
1 M_Out32(10000) = P1.X * 1000      'Multiplikation des X-Koordinatenwerts von P1 mit
                                    1000 und als 32-Bit-Wert der Ausgangssignalnummer
                                    10000 schreiben (Integer-Wert)
2 *ack_wait
3 If M_In(7) = 0 Then *ack_wait     'Warten, bis Eingangssignal 7 einschaltet
                                    (Unterbrechung).
4 P1.Y = M_In32(10100)/1000         'Eingabe des Ergebniswerts der Division des 32-Bit-
                                    Datenwerts der Eingangssignalnr. 10100 mit dem Wert
                                    1000 in die Y-Komponente der Positionsvariable P1
                                    (Dieses Beispiel setzt voraus, dass das externe Signal
                                    ursprünglich ein Real-Wert war, danach mit 1000
                                    multipliziert und in einen Integer-Wert umgewandelt
                                    wurde.)
```

Erläuterung

- Die Variable ermöglicht den Schreib- und Lesezugriff auf externe Ausgangssignale als numerischer Wert mit einer Länge von 32 Bit.
- Die Daten werden von der festgelegten Signalnummer im 32-Bit-Format ausgegeben.
- Die Taktzeit zur Aktualisierung der externen Ausgangssignale kann durch die Einstellung des Parameters SYNCIO auf „High-Speed" verkürzt werden. Die entsprechende Einstellung ist in Abschnitt 9.3 beschrieben.

ACHTUNG:

Unterbrechen Sie immer die Signale, um eine Synchronisation auszulösen. Andernfalls können durch falsche oder unvollständig übertragene Signale Fehlfunktionen auftreten.

Bit-Anzahl	Konstanten			Numerische Variablen				Sonstige Variablen		
	Numerischer Wert ①	Binäre Zahl	Hexadezimale Zahl	Integer	Integer-Wert mit Doppelwort-Genauigkeit	Real-Wert mit einfacher Genauigkeit	Real-Wert mit doppelter Genauigkeit	Position ①②	Gelenk ①②	Zeichenkette
	Bsp.: 12	Bsp.: &B1100	Bsp.: &HC	Bsp.: M1%	Bsp.: M1&	Bsp.: M1!	Bsp.: M1#	Bsp.: P1.X	Bsp.: J1.J1	Bsp.: C1$
M_Out32	●	●	●	●	●	●	●	●	○	○

Tab. 7-10: *Numerischer Wert 1*

●: Verfügbar

○: Nicht verfügbar

① Real-Wert wird abgerundet

② Wenn die Variable den Wert eines Winkels hat, wird diese mit der Einheit „Radiant" verarbeitet. (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Daher ist die Zuordnung der Signalnummer sehr schwierig. Zur Anzeige wird der Wert in „Grad" umgewandelt und es wird der gleiche Wert, wie der Einstellwert dargestellt.
Beispiel:
Es wird der Wert „0" verarbeitet, obwohl für P1.A zur Festlegung der Eingangssignalnummer der Wert „8" eingestellt wurde (z. B. über Eingabe mit der Tastatur der Teaching Box usw.). Der Wert „8" (Grad) wird in „0,14" (Radiant) umgewandelt, was nach Abrundung den Wert „0" ergibt. Da die Einheit der Komponenten X, Y und Z der Positionsvariable „mm" ist, existiert dieser Zustand nicht.

Bit-Anzahl	Konstanten			Numerische Variablen				Sonstige Variablen		
	Numerischer Wert Bsp.: 12	Binäre Zahl Bsp.: &B1100	Hexadezimale Zahl Bsp.: &HC	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit ① Bsp.: M1!	Real-Wert mit doppelter Genauigkeit ① Bsp.: M1#	Position Bsp.: P1.X	Gelenk Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_Out32	● ①	● ①	● ①	● ①	●	●	● ②	●	●	○

Tab. 7-11: *Numerischer Wert*

●: Verfügbar

○: Nicht verfügbar

① Beachten Sie unbedingt den folgenden Hinweis:

ACHTUNG:

Der numerische Wert einer 16 Bit breiten Binärzahl (–32768 to +32767) wird bei der Zuweisung durch eine Konstante als negativer Wert verarbeitet, wenn das 16te Bit eingeschaltet ist. Beachten Sie, dass dadurch auch die höherwertigen 16 Bit eingeschaltet werden. (Das Vorzeichenbit wird erweitert.)
Wird beispielsweise M_Out32 der Wert „–32768 (&B1000000000000000)" zugewiesen, dann wird „&B11111111111111111000000000000000" ausgegeben.
Abhilfe:
Übertragen Sie die Konstante wie unten gezeigt zuerst in die Integer-Variable mit doppelter Wortgenauigkeit und anschließend in die Systemstatusvariable M_YDevD, um den Wert &B00000000000000001000000000000000 (Binärwert) auszugeben.
1 M1& = 32768
2 M_YDevD(&H20) = M1&

② Der Ausgabebereich für numerische Werte liegt von –2147483648 bis 2147483647.

Bit-Anzahl	Numerische Variablen				Sonstige Variablen		
	Integer Bsp.: M1%	Integer-Wert mit Doppelwort-Genauigkeit Bsp.: M1&	Real-Wert mit einfacher Genauigkeit ① Bsp.: M1!	Real-Wert mit doppelter Genauigkeit ① Bsp.: M1#	Position ① Bsp.: P1.X	Gelenk ① Bsp.: J1.J1	Zeichenkette Bsp.: C1$
M_Out32	●	●	●	●	●	●	○

Tab. 7-12: *Numerischer Wert*

●: Verfügbar

○: Nicht verfügbar

① Wenn die Variable den Wert eines Winkels hat, wird diese mit der Einheit „Radiant" verarbeitet. Der aktuelle 32-Bit-Ausgabewert wird ohne Umwandlung in einen numerischen Radiant-Wert gespeichert. (Dies gilt für alle Komponenten A, B und C der Positionsvariable und alle Komponenten der Gelenkvariable.) Zur Anzeige wird der Wert in „Grad" umgewandelt.

Steht in Beziehung zu folgenden Variablen:

M_Out/M_Outb/M_Out8/M_Outw/M_Out16

Steht in Beziehung zu folgenden Befehlen:

Def IO

Steht in Beziehung zu folgenden Parametern:

SYNCIO

7.2.38 M_PI

Funktion: Kreiszahl lesen

Die Variable enthält den Wert der Kreiszahl π (3,14159265358979).

Eingabeformat

Bsp.: <Numerische Variable> = M_PI

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
1 M1 = M_PI                    'Überträgt den Wert der Kreiszahl (3,1415926)
                               in die numerische Variable M1
```

Erläuterung

- Die Variable ist vom Typ Real.
- Die Variable kann ausschließlich gelesen werden.

7.2.39 M_Psa

Funktion: Programmwählbarkeit lesen

Die Variable zeigt an, ob ein Programm in einem Programmplatz ausgewählt werden kann.
1: Programm kann ausgewählt werden
0: Programm kann nicht ausgewählt werden (Programm unterbrochen)

Eingabeformat

Bsp.: <Numerische Variable> = M_Psa [(<Numerischer Ausdruck>)]

<Numerische Variable> Legt eine numerische Variable fest

<Numerischer Ausdruck> Legt die Programmplatznummer fest
$1 \leq$ Programmplatznummer ≤ 32
Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1 M1 = M_Psa(2)                'Überträgt den Status der Programmwählbarkeit von
                               Programmplatz 2 in die numerische Variable M1
```

Erläuterung

- Die Variable ermöglicht eine Abfrage, ob ein Programm über einen Programmplatz ausgewählt werden kann.
- Die Variable kann ausschließlich gelesen werden.

7.2.40 M_Ratio

Funktion: Annäherung an die Zielposition lesen

Die Variable ermöglicht während der Roboterbewegung eine Überwachung des bereits zur Zielposition zurückgelegten Verfahrweges im Bereich von 0 bis 100 %.

Eingabeformat

Bsp.: <Numerische Variable> = M_RATIO [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1  Mov P1 WthIf M_Ratio > 80, M_Out(1) = 1     'Position P1 mittels Gelenk-Interpolation
                                               anfahren und Ausgangsbit 1 auf „1“ setzen,
                                               sobald 80 % des Verfahrweges zurückgelegt
                                               worden sind
```

Erläuterung

- Die Variable wird z. B. dazu verwendet, an spezifischen Positionen des Verfahrweges bestimmte Prozeduren auszuführen.
- Die Variable kann ausschließlich gelesen werden.

7.2.41 M_RDst

Funktion: Abstand zur Zielposition lesen

Die Variable ermöglicht während der Roboterbewegung eine Überwachung des Abstands zur Zielposition in Millimetern.

Eingabeformat

Bsp.: <Numerische Variable> = M_RDst [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1  Mov P1 WthIf M_RDst < 10, M_Out(1) = 1    'Position P1 mittels Gelenk-Interpolation
                                             anfahren und Ausgangsbit 1 auf „1“ setzen,
                                             sobald der Abstand zur Zielposition kleiner als
                                             10 mm ist
```

Erläuterung

- Die Variable wird z. B. dazu verwendet, an spezifischen Positionen des Verfahrweges bestimmte Prozeduren auszuführen.
- Die Variable kann ausschließlich gelesen werden.

7.2.42 M_Run

Funktion: Programmstatus lesen

Die Variable enthält den Status eines Programms.
1: Ausführung
0: keine Ausführung (unterbrochen oder gestoppt)

Eingabeformat

Bsp.: <Numerische Variable> = M_Run [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1  M1 = M_Run(2)          'Überträgt den Status des Programms in Programm-
                          platz 2 in die numerische Variable M1
```

Erläuterung

- Der Wert der Variablen ist „1“, wenn das Programm ausgeführt wird und „0“, wenn das Programm unterbrochen oder gestoppt ist.
- Verwenden Sie die Variable M_Run oder M_Wai, um den Stoppzustand eines Programmes zu überprüfen (falls die aktuell ausgeführte Zeile die oberste Programmzeile ist).
- Die Variable kann ausschließlich gelesen werden.

7.2.43 M_SetAdl

Funktion: Beschleunigungs-/Abbremszeit jeder Achse einstellen

Die Variable ermöglicht eine Einstellung der Beschleunigungs-/Abbremszeit jeder einzelnen Gelenkachse. Dadurch kann die Motorbelastung einer hoch belasteten Achse gezielt reduziert werden. Die Variable ist auch beim Betrieb mit optimaler Beschleunigung/Abbremsung (Oadl ON) wirksam.
Die Einstellung der Geschwindigkeit des gesamten Roboters über die Befehle Ovrd bzw. Spd oder der Beschleunigungs-/Abbremszeit über den Befehl Accel kann in vielen Fällen zu einer unnötigen Vergrößerung der Taktzeiten führen. Der Initialisierungswert ist im Parameter JADL festgelegt.

Die Variable M_SetAdl kann nur mit den Robotermodellen RV-2SDB/3SDB/6SD/6SDL/12SD/12SDL, RH-3SDHR, RV-2SQB/3SQB/6SQ/6SQL/12SQ/12SQL und RH-3SQHR verwendet werden.

Eingabeformat

Bsp.: M_SetAdl (<Achsennummer>) = <Numerische Variable>

<Achsennummer>	Legt die Achsennummer fest 1 ≤ Achsennummer ≤ 8
<Numerische Variable>	Legt eine die Standard-Beschleunigungs-/Abbremszeit im Bereich von 1 bis 100 % fest Der Initialisierungswert ist im Parameter JADL festgelegt.

Programmbeispiel

```
1  Accel 100,50                'Abbremsung auf 50 % reduzieren
2  If M_LdFact(2) > 90 Then    'Bei Überschreitung des Lastverhältnisses von 90 %,
                               gehe zur folgenden Zeile
3     M_SetAdl(2) = 70         'Beschleunigung der Achse J2 auf 70 % (100 % × 70 %)
                               und Abbremsung auf 35 % (50 % × 70 %) verringern
4  EndIf
5  Mov P1                      'Position P1 mittels Gelenk-Interpolation anfahren
6  Mov P2                      'Position P2 mittels Gelenk-Interpolation anfahren
7  M_SetAdl(2) = 100           'Beschleunigung der Achse J2 wieder auf 100 % und
                               Abbremsung auf 50 % einstellen
8  Mov P3                      'Position P3 mittels Gelenk-Interpolation anfahren
9  Accel 100,100               'Beschleunigung/Abbremsung wieder auf Standard-
                               last einstellen
10 Mov P4                      'Position P4 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Variable ermöglicht eine Einstellung der Beschleunigungs-/Abbremszeit für jede Gelenkachse. Dabei entsprechen 100 % der kürzesten Beschleunigungs-/Abbremszeit.
- Mit Hilfe der Variablen kann die Beschleunigungs-/Abbremszeit und somit die Belastung einer Achse, die eine Überlast- oder Überhitzungsfehlermeldung hervorruft, gezielt eingestellt werden.
- Die Einstellung der Variablen wirkt sich gleichzeitig auf die Beschleunigungs- und Bremszeit aus.
- Wird die Variable im Betrieb mit optimaler Beschleunigung/Abbremsung gemeinsam mit dem Befehl Accel verwendet, ergeben sich die Werte für die Beschleunigung-/Abbremsung aus der Multiplikation der einzelnen Faktoren (siehe auch Abschn. 6.3.57).
- Die Einstellung der Beschleunigungs-/Bremszeit erfolgt über den Befehl Accel. Da über die Variable M_SetAdl eine Einstellung der einzelnen Achsen erfolgt und der berechnete Wert der Beschleunigungs-/Bremszeit auch von der Motorbelastung abhängt, kann der Istwert der Beschleunigungs-/Bremszeit etwas vom vorher ermittelten Sollwert abweichen.

Steht in Beziehung zu folgenden Befehlen:

Accel, Ovrd, Spd

Steht in Beziehung zu folgenden Variablen:

M_LdFact

7.2.44 M_SkipCq

Funktion: Skip-Befehlsausführung lesen

Die Variable zeigt die Ausführung eines SKIP-Befehls an.
1: Skip-Befehl wurde ausgeführt.
0: Skip-Befehl wurde nicht ausgeführt.

Eingabeformat

Bsp.: <Numerische Variable> = M_SkipCq [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1  Mov P1 WthIf M_In(10) = 1,Skip     'Fährt Position 1 mittels Gelenk-Interpolation an und
                                      unterbricht die Roboterbewegung, wenn das Ein-
                                      gangsbit Nummer 10 gleich 1 wird
                                      Die Programmsteuerung springt in die nächste Zeile.
2  If M_SkipCq = 1 Then GoTo *Lskip   'Springt bei Ausführung der Skip-Anweisung zur Marke
                                      Lskip
   :
10 *Lskip                             'Sprungmarke Lskip festgelegt
```

Erläuterung

- Mit Hilfe der Variablen kann geprüft werden, ob ein Skip-Befehl ausgeführt worden ist.
- Nach der Prüfung wird der Skip-Status zurückgesetzt. (Die Variable wird auf „0“ gesetzt.) Soll der Status erhalten bleiben, ist der Wert in eine numerische Variable zu übertragen.
- Die Variable kann ausschließlich gelesen werden.

7.2.45 M_Spd/M_NSpd/M_RSpd

Funktion: Geschwindigkeit lesen

Die Variable ermöglicht eine Überwachung der Geschwindigkeit während der Linear- und Gelenk-Interpolation.

M_Spd: aktuell eingestellte Geschwindigkeit
M_NSpd: Systemstandardwert (optimale Geschwindigkeit)
M_RSpd: aktuelle Geschwindigkeit

Eingabeformat

Bsp.: <Numerische Variable> = M_SPD [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_NSPD [(<Numerischer Ausdruck>)]

Bsp.: <Numerische Variable> = M_RSPD [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1 M1 = M_Spd          'Überträgt den zuletzt eingestellten Geschwindigkeitswert in die numerische Variable M1
2 Spd M_NSpd          'Setzt die Geschwindigkeit auf den Systemstandardwert zur Steuerung mit optimaler Geschwindigkeit
```

Erläuterung

- Die Variable M_RSPD enthält die aktuelle Verfahrgeschwindigkeit des Roboters. Ist die Servoversorgung abgeschaltet, enthält die Variable den Wert der zurückgemeldeten Geschwindigkeit. In diesem Moment kann der Wert, auch im Stoppzustand, zwischen –0,01 und +0,01 schwanken.
- Die Variablen können zur geschwindigkeitsabhängigen Steuerung von Multitasking-Programmen oder in Kombination mit den angehängten Anweisungen Wth und WthIf verwendet werden.
- Die Variablen können ausschließlich gelesen werden.

7.2.46 M_Svo

Funktion: Status der Servoversorgung lesen

Die Variable zeigt den Status der Servoversorgungsspannung an.
1: Servoversorgung eingeschaltet
0: Servoversorgung ausgeschaltet

Eingabeformat

Bsp.: <Numerische Variable> = M_SVO [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  M1 = M_Svo(1)          'Überträgt den Status der Servoversorgung von
                          Mechanismus 1 in die numerische Variable M1
```

Erläuterung

- Mit Hilfe der Variablen kann der Status der Servoversorgungsspannung eines Mechanismus geprüft werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.47 M_Timer

Funktion: Zeitdauer schreiben/lesen

Die Bezugszeit wird in Millisekunden gemessen. Die Variable dient zur genauen Erfassung von Vorgangsdauern und zur genauen Zeitmessung.

Eingabeformat

Bsp.: <Numerische Variable> = M_Timer (<Numerischer Ausdruck>)

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Timer-Nummer zwischen 1 und 8 fest

Programmbeispiel

```
1 M_Timer(1) = 0          'Setzt Timer 1 auf „0“
2 Mov P1                  'Position P1 mittels Gelenk-Interpolation anfahren
3 Mov P2                  'Position P2 mittels Gelenk-Interpolation anfahren
4 M1 = M_Timer(1)         'Schreibt die Dauer für die Verfahrbewegung von P1
                          nach P2 in ms in die numerische Variable M1
                          (Bsp.: Eine Zeit von 5,346 s entspricht einem Wert von
                          M1 von 5346)
5 M_Timer(1) = 1.5        'Setzt Timer 1 auf 1,5 s
```

Erläuterung

- Wird der Variablen ein Wert zugewiesen, erfolgt die Zuweisung in Sekunden.
- Die Auflösung in Millisekunden ermöglicht eine hochgenaue Zeiterfassung.

7.2.48 M_Tool

Funktion: Werkzeugnummer schreiben/lesen

Ein Zugriff auf die Werkzeugdaten der verschiedenen Werkzeuge kann über die Parameter MEXTL1 bis 16 oder die Variable M_Tool erfolgen.

Eingabeformat

Werkzeugnummer lesen

Bsp.: <Numerische Variable> = M_Tool [(<Mechanismusnummer>)]

Werkzeugnummer schreiben

Bsp.: M_Tool [(<Mechanismusnummer>)] = <Numerischer Ausdruck>

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.
<Numerischer Ausdruck>	Legt die Werkzeugnummer zwischen 1 und 16 fest Der Wert muss in Klammern angegeben werden.

Programmbeispiel

Festlegung der Werkzeugdaten

```
1  Tool (0,0,100,0,0,0)        'Werkzeugdaten (0,0,100,0,0,0) festlegen und in die
                                Variable MEXTL übertragen
2  Mov P1                      'Position P1 mittels Gelenk-Interpolation anfahren
3  M_Tool = 2                  'Werkzeugdaten von Werkzeug 2 aktivieren (MEXTL2)
4  Mov P2                      'Position P2 mittels Gelenk-Interpolation anfahren
```

Umschalten der Werkzeugnummer

```
1  If M_In(900) = 1 Then       'Werkzeugdaten über Handeingangssignal
                               umschalten
2     M_Tool = 1               'Werkzeugdaten von Werkzeug 1 aktivieren
3  Else
4     M_Tool = 2               'Werkzeugdaten von Werkzeug 2 aktivieren
5  EndIf
6  Mov P1                      'Position P1 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Werkzeugdaten werden in den Parametern MEXTL1, MEXTL2, MEXTL3 bis MEXTL16 eingestellt und in den Parameter MEXTL übertragen.
- Die Werkzeuge 1 bis 16 entsprechen den Parametern MEXTL1 bis 16.
- Beim Lesezugriff auf die Variable M_TOOL wird die aktuelle Werkzeugnummer übertragen.
- Ist der eingelesene Wert 0, bedeutet dies, dass die aktuellen Werkzeugdaten nicht über die Parameter MEXTL1 bis 16 aktiviert wurden.
- Die Auswahl der Werkzeugdaten kann auch über die Teaching Box erfolgen (siehe auch Abschn. 3.5).

Steht in Beziehung zu folgenden Befehlen:

Tool

Steht in Beziehung zu folgenden Parametern:

MEXTL, MEXTL1 bis 16

7.2.49 M_Uar

Funktion: Aufenthalt im benutzerdefinierten Bereich prüfen

Die Variable zeigt an, ob der Mechanismus sich innerhalb des benutzerdefinierten Bereichs befindet. Dabei entsprechen die Bits 0 bis 7 den benutzerdefinierten Bereichen 1 bis 8.
1: innerhalb des benutzerdefinierten Bereichs
0: außerhalb des benutzerdefinierten Bereichs

Eingabeformat

Bsp.: <Numerische Variable> = M_Uar [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  M1 = M_Uar(1)        'M1 zeigt an, ob der Mechanismus sich innerhalb oder
                        außerhalb des benutzerdefinierten Bereichs befindet
                        Der Wert „4“ zeigt z. B. an, dass der Roboter sich im
                        benutzerdefinierten Bereich 3 befindet.
```

Erläuterung

- Eine detaillierte Beschreibung zur Anwendung benutzerdefinierter Bereiche finden Sie in Abschn. 9.9.
- Die Variable kann ausschließlich gelesen werden.

7.2.50 M_Uar32

Funktion: Aufenthalt im benutzerdefinierten Bereich prüfen

Die Variable zeigt an, ob der Mechanismus sich innerhalb des benutzerdefinierten Bereichs befindet. Dabei entsprechen die Bits 0 bis 31 den benutzerdefinierten Bereichen 1 bis 32.
1: innerhalb des benutzerdefinierten Bereichs
0: außerhalb des benutzerdefinierten Bereichs

Eingabeformat

Bsp.: <Numerische Variable> = M_Uar32 [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1 Def Long M1
2 M1& = M_Uar32(1) AND &H00080000     'Ergebnis für den Bereich 20 in M1 übertragen.
3 If M1& <> 0 Then M_Out(10) = 1      'Ausgangssignal 10 schaltet ein, wenn sich der
                                       Mechanismus im Bereich 20 befindet
```

Erläuterung

- Eine detaillierte Beschreibung zur Anwendung benutzerdefinierter Bereiche finden Sie in Abschn. 9.9.
- Es erfolgt eine Fehlermeldung, wenn beim Eingabeformat für die <numerische Variable> eine 16-Bit-Integer-Zahl verwendet wird und dabei der max. Wertebereich überschritten wird. Verwenden Sie in diesem Fall für die <numerische Variable> eine 32-Bit-Integer-Zahl.
- Es wird der Bereich überprüft, für den in Parameter AREAnAT (n = 1–32) der Wert „1“ (Signalausgabe) eingestellt wurde.
- Bei der dezimalen Darstellung erhält man bei Vergleichsoperationen oder logischen Rechenoperationen ein negatives Ergebnis, wenn Bit 31 den Wert „1“ hat. Um dies zu verhindern wird eine hexadezimale Darstellung empfohlen.
- Eine entsprechende Zuordnung der benutzerdefinierbare Bereiche zeigt Tab. 7-13.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

M_Uar

Bit	Bereich	Dezimaler Wert	Hexadezimaler Wert
0	1	1	&H00000001
1	2	2	&H00000002
2	3	4	&H00000004
3	4	8	&H00000008
4	5	16	&H00000010
5	6	32	&H00000020
6	7	64	&H00000040
7	8	128	&H00000080
8	9	256	&H00000100
9	10	512	&H00000200
10	11	1024	&H00000400
11	12	2048	&H00000800
12	13	4096	&H00001000
13	14	8192	&H00002000
14	15	16384	&H00004000
15	16	32768	&H00008000

Bit	Bereich	Dezimaler Wert	Hexadezimaler Wert
16	17	65536	&H00010000
17	18	131072	&H00020000
18	19	262144	&H00040000
19	20	524288	&H00080000
20	21	1048576	&H00100000
21	22	2097152	&H00200000
22	23	4194304	&H00400000
23	24	8388608	&H00800000
24	25	16777216	&H01000000
25	26	33554432	&H02000000
26	27	67108864	&H04000000
27	28	134217728	&H08000000
28	29	268435456	&H10000000
29	30	536870912	&H20000000
30	31	1073741824	&H40000000
31	32	–2147483648	&H80000000

Tab. 7-13: *Benutzerdefinierbare Bereiche der Variable M_Uar32*

Beispiel ▽ Wenn sich der Mechanismus in Bereich 5 und Bereich 10 befindet, dann muss die Variable M_Uar32 mit dem hexadezimalen Wert &H00000010 (Bereich 5) und &H00000200 (Bereich 10) maskiert werden (AND-Verknüpfung).

△

7.2.51 M_Wai

Funktion: Wartestatus lesen

Die Variable zeigt an, ob sich das Programm im ausgewählten Programmplatz im Wartestatus befindet.
1: Wartestatus (Das Programm ist unterbrochen.)
0: kein Wartestatus (Das Programm wird abgearbeitet oder befindet sich im Stoppzustand.)

Eingabeformat

Bsp.: <Numerische Variable> = M_WAI [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest 1 ≤ Programmplatznummer ≤ 32 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

```
1  M1 = M_Wai(1)        'M1 zeigt an, ob sich das Programm im Programmplatz 1 im Wartestatus befindet
```

Erläuterung

- Mit Hilfe der Variablen M_Wai kann der Wartestatus eines Programms überprüft werden.
- Verwenden Sie die Variable M_Run oder M_Wai, um den Stoppzustand eines Programmes zu überprüfen (falls der aktuell ausgeführte Programmschritt der oberste Programmschritt ist).
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

M_Wupov, M_Wuprt, M_Wupst

7.2.52 M_Wupov

Funktion: Übersteuerung im Warmlaufbetrieb lesen

Die Variable enthält den Wert der Übersteuerung in Prozent, der dem Geschwindigkeitssollwert im Warmlaufbetrieb zur Reduzierung der Betriebsgeschwindigkeit überlagert wird.

Eine detaillierte Beschreibung des Warmlaufbetriebs finden Sie in Abschn. 9.22.

Eingabeformat

Bsp.: <Numerische Variable> = M_Wupov [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  M1 = M_Wupov(1)        'Überträgt den Übersteuerungswert für den
                           Warmlaufbetrieb von Mechanismus 1 in die
                           numerische Variable M1
```

Erläuterung

- Mit Hilfe der Statusvariablen kann der Übersteuerungswert im Warmlaufbetrieb (Betrieb mit verminderter Geschwindigkeit), der dem Geschwindigkeitssollwert im Warmlaufbetrieb zur Reduzierung der Betriebsgeschwindigkeit überlagert wird, geprüft werden.
- Ist der Warmlaufbetrieb deaktiviert, der [MODE]-Umschalter am Steuergerät auf „TEACH“ eingestellt oder der Mechanismus gesperrt, ist der Wert auf 100 gesetzt.
- Bei Umschaltung vom Normal- in den Warmlaufbetrieb oder einer Aktivierung des Warmlaufbetriebs direkt nach dem Einschalten, wird der Wert, der im ersten Element des Parameters WUPOvrd eingestellt ist, als Startwert der Variablen verwendet. Mit fortschreitender Betriebsdauer des Roboters vergrößert sich der Wert der Variablen M_Wupov. Bei Beendigung des Warmlaufbetriebs wird der Wert der Variablen M_Wupov auf 100 gesetzt.
- Die aktuelle Übersteuerung im Warmlaufbetrieb ergibt sich wie folgt:

Gelenk-Interpolation	=	Übersteuerungswert der T/B oder des Steuergeräts	x	Einstellwert des Ovrd-Befehls	x	Einstellwert des JOvrd-Befehls	=	Übersteuerungswert für den Warmlaufbetrieb
Linear-Interpolation	=	Übersteuerungswert der T/B oder des Steuergeräts	x	Einstellwert des Ovrd-Befehls	x	Einstellwert des Spd-Befehls	=	Übersteuerungswert für den Warmlaufbetrieb

- Die Variable kann ausschließlich gelesen werden.

7.2.53 M_Wuprt

Funktion: Restzeit einer Achse im Warmlaufbetrieb

Die Variable enthält die Restzeit in Sekunden, in der eine Achse im Warmlaufbetrieb verfahren wird, bis der Warmlaufbetrieb beendet ist.

Eine detaillierte Beschreibung des Warmlaufbetriebs finden Sie in Abschn. 9.22.

Eingabeformat

Bsp.: <Numerische Variable> = M_Wuprt [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

1 M1 = M_Wuprt(1)	'Überträgt die Restzeit der für den Warmlaufbetrieb festgelegten Achse von Mechanismus 1, in der die Achse bis zur Beendigung des Warmlaufbetriebs im Warmlaufbetrieb verfahren wird, in die numerische Variable M1

Erläuterung

- Mit Hilfe der Statusvariablen kann die Restzeit, die eine über den Parameter WUPAXIS festgelegte Achse für den Warmlaufbetrieb (Betrieb mit verminderter Geschwindigkeit) im Warmlaufbetrieb verfahren wird, bis der Warmlaufbetrieb beendet ist, überprüft werden.
- Bei deaktiviertem Warmlaufbetrieb ist der Wert der Variablen „0“.
- Bei Umschaltung vom Normal- in den Warmlaufbetrieb oder einer Aktivierung des Warmlaufbetriebs direkt nach dem Einschalten, wird der Wert, der im ersten Element des Parameters WUPTIME eingestellt ist, als Startwert der Variablen verwendet. Mit fortschreitender Betriebsdauer des Roboters verkleinert sich der Wert der Variablen M_Wuprt. Erreicht der Wert „0“, wird der Warmlaufbetrieb beendet. Sind mehrere Achsen für einen Warmlaufbetrieb festgelegt, enthält die Variable den Wert der Achse mit der kürzesten Betriebsdauer.

Beispiel ▽ Die Achse A arbeitet bis zur Beendigung des Warmlaufbetriebs noch 20 s (bei M_Wuprt = 20) im Warmlaufbetrieb. Wird in dieser Zeit eine Achse B nach einer Betriebspause vom Normal- in den Warmlaufbetrieb umgeschaltet, so ist dies die Achse mit der kürzesten Betriebdauer (Betriebsdauer = 0 s). Die Restzeit der Achse B bis zur Beendigung des Warmlaufbetriebs beträgt 60 s (Werkseinstellung = 60 s). Somit wird dieser Wert in die Statusvariable übertragen (M_Wuprt = 60 s).

△

- Die Variable kann ausschließlich gelesen werden.

7.2.54 M_Wupst

Funktion: Zeit bis zur Wiederholung des Warmlaufbetriebs

Die Variable enthält die Zeit in Sekunden, bis nach Beendigung eines Warmlaufbetriebs ein neuer Warmlaufbetrieb aktiviert wird.

Eine detaillierte Beschreibung des Warmlaufbetriebs finden Sie in Abschn. 9.22.

Eingabeformat

Bsp.: <Numerische Variable> = M_Wupst [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

1 M1 = M_Wupst(1)	'Überträgt die Zeit, bis für Mechanismus 1 nach Beendigung eines Warmlaufbetriebs ein erneuter Warmlaufbetrieb aktiviert wird, in die numerische Variable M1

Erläuterung

- Mit Hilfe der Statusvariablen kann die Zeit, bis eine über den Parameter WUPAXIS festgelegte Achse nach Beendigung des Warmlaufbetriebs erneut im Warmlaufbetrieb (Betrieb mit verminderter Geschwindigkeit) verfahren wird, überprüft werden.
- Bei deaktiviertem Warmlaufbetrieb entspricht der Wert der Statusvariablen der im zweiten Element des Parameters WUPTIME (Wiederholschwelle) festgelegten Zeit.
- Beim Verfahren einer Achse nach Beendigung des Warmlaufbetriebs wird der im zweiten Element des Parameters WUPTIME (Wiederholschwelle) festgelegte Wert als Anfangszeit in die Statusvariable M_Wupst übertragen. Während der Betriebspause der für den Warmlaufbetrieb festgelegten Achse verringert sich der Wert. Erreicht der Wert „0“, wird ein erneuter Warmlaufbetrieb aktiviert.
- Sind mehrere Achsen für einen Warmlaufbetrieb festgelegt, wird der Wert der Achse mit der längsten Betriebspause in die Statusvariable übertragen.
- Die Variable kann ausschließlich gelesen werden.

7.2.55 P_Base/P_NBAase

Funktion: Basis-Konvertierungsdaten lesen

Die Variable enthält Werte, die auf die Basis-Konvertierungsdaten bezogen sind.

P_Base: aktuell eingestellte Basis-Konvertierungsdaten
P_NBase: Standardwert der Basis-Konvertierungsdaten (0, 0, 0, 0, 0, 0) (0, 0)

Eingabeformat

Bsp.: <Positionsvariable> = P_Base [(<Mechanismusnummer>)]

Bsp.: <Positionsvariable> = P_NBase

<Positionsvariable> Legt eine Positionsvariable fest

<Mechanismusnummer> Legt die Mechanismusnummer fest
$1 \leq$ Mechanismusnummer ≤ 3
Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1 P1 = P_Base      'Überträgt die aktuell eingestellten Basis-
                   Konvertierungsdaten in die Positionsvariable P1
2 Base P_NBase     'Setzt die Basis-Konvertierungsdaten auf den
                   Standardwert zurück
```

Erläuterung

- Der Standardwert der Basis-Konvertierungsdaten ist auf (0, 0, 0, 0, 0, 0) (0, 0) gesetzt.
- Beachten Sie, dass eine Änderung der Basiskonvertierungsdaten die geteachten Positionen beinflussen kann.
- Eine Änderung des Basiskoordinatensystems erfolgt über den Base-Befehl. Fehlerhaft eingestellte Basis-Konvertierungsdaten können zu undefinierten Aktivitäten des Roboters führen. Stellen Sie daher die Werte besonders gewissenhaft ein.
- Die Variable kann ausschließlich gelesen werden.

7.2.56 P_ColDir

Funktion: Verfahrwegrichtung bei einem Zusammenstoß lesen

Die Variable enthält die Richtung des Verfahrweges bei Ansprechen der Kollisionsüberwachung.

Eingabeformat

Bsp.: <Positionsvariable> = P_ColDir [(<Mechanismusnummer>)]

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

Bei einem Zusammenstoß erfolgt der Aufruf eines Interrupt-Prozesses

```
1  Def Act 1,M_ColSts(1) = 1 GoTo *HOME,S   'Definiert bei einem Zusammenstoß einen
                                            Unterprogrammsprung zur Marke HOME
2  Act 1 = 1                                'Interrupt 1 freigeben
3  ColCck On,NOErr                          'Kollisionsüberwachung ohne Fehlerausgabe
                                            aktivieren
4  Mov P1
5  Mov P2                                   'Erfolgt während der Ausführung der Zeilen
                                            4 bis 7 ein Zusammenstoß, wird der Interrupt-
                                            Prozess ausgeführt
6  Mov P3
7  Mov P4
8  Act 1 = 0                                'Interrupt 1 sperren
   :
10 *HOME                                    'Interrupt-Prozess bei einem Zusammenstoß
11 ColChk Off                               'Kollisionsüberwachung deaktivieren
12 Servo On                                 'Schaltet die Servospannung ein
13 PESC = P_ColDir(1)*(–2)                  'Abstand der Ausweichposition in
                                            entgegengesetzter Richtung festlegen
14 PDst = P_Fbc(1) + PESC                   'Ausweichposition festlegen
15 Mvs PDst                                 'Ausweichposition mittels Linear-Inter-
                                            polation anfahren
16 Error 9100                               'Benutzerdefinierten, leichten Fehler
                                            ausgeben
```

Erläuterung

- Die Variable dient zur Festlegung der Bewegungsrichtung des Roboters für die automatische Anfahrt einer Ausweichposition nach einem Zusammenstoß.
- Die Berechnung der Bewegungsrichtung im Augenblick des Zusammenstoßes erfolgt indem die Komponente der Position der Hauptbewegungsrichtung auf „1“ gesetzt und eine Anpassung der anderen Komponenten im gleichen Verhältnis durchgeführt wird.
  Ist das Verhältnis z. B. X/Y = 2/1 ergibt sich P_ColDir = (1,0.5,0,0,0,0)(0,0).
- Stellungsdaten und Stellungsmerker sind auf „0“ gesetzt: (*,*,*,0,0,0,0,0)(0,0).
- Der berechnete Wert bei einem Zusammenstoß bleibt bis zum nächsten Zusammenstoß erhalten.
- Spricht die Kollisionsüberwachung durch Einwirkung äußerer Kräfte an, während der Roboter im Stillstand ist, werden die Werte für alle Achsen auf „0“ gesetzt.
- Da die Berechnung der Bewegungsrichtung mit Bezug auf die Zielposition eines Bewegungsbefehls erfolgt, werden alle Komponenten der Variablen bei einem Zusammenstoß nahe der Zielposition auf „0.0“ gesetzt.
- Die Variable kann ausschließlich gelesen werden.
- Bei Robotern, die nicht über die Funktion der Kollisionsüberwachung verfügen, sind alle Komponenten der Variablen auf „0.0“ gesetzt.

Steht in Beziehung zu folgenden Befehlen:

ColChk, ColLvl

Steht in Beziehung zu folgenden Variablen:

M_ColSts, J_ColMxl

7.2.57 P_Curr

Funktion: aktuelle Position lesen

Die Variable enthält die aktuelle Postion (X, Y, Z, A, B, C, L1, L2) (FL1, FL2).

Eingabeformat

Bsp.: <Positionsvariable> = P_Curr [(<Mechanismusnummer>)]

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  Def Act 1, M_In(10) = 1 GoTo *LACT     'Definiert einen Sprung zur Marke LACT, wenn
                                          der Status des allgemeinen Eingangssignals
                                          Nummer 10 = EIN ist
2  Act 1 = 1                              'Interrupt 1 freigeben
3  Mov P1                                 'Position P1 mittels Gelenk-Interpolation anfahren
4  Mov P2                                 'Position P2 mittels Gelenk-Interpolation anfahren
5  Act 1 = 0                              'Interrupt 1 sperren
   :
100  *LACT
101  P100 = P_Curr                        'Lädt die aktuelle Position in die Positionsvariable
                                          P100, wenn ein Interrupt anliegt
102  Mov P100,–100                        'Position anfahren, die 100 mm in Werkzeuglängs-
                                          richtung von der Position P100 entfernt ist
103  End                                  'Programmende
```

Erläuterung

- Die Variable überträgt die aktuelle Position.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

J_Curr

7.2.58 P_Fbc

Funktion: aus der Servorückmeldung ermittelte Position lesen

Die Variable enthält die aus den Daten der Servorückmeldung ermittelte aktuelle Postion (X, Y, Z, A, B, C, L1, L2) (FL1, FL2).

Eingabeformat

Bsp.: <Positionsvariable> = P_Fbc [(<Mechanismusnummer>)]

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  P1 = P_Fbc          'Überträgt die aus den Daten der Servorückmeldung
                        ermittelte Position in die Positionsvariable P1
```

Erläuterung

- Die Variable überträgt die aus der Servorückmeldung ermittelte aktuelle Position.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Befehlen:

Torq

Steht in Beziehung zu folgenden Variablen:

J_Fbc/J_AmpFbc, M_Fbd

7.2.59 P_Safe

Funktion: Rückzugspunkt lesen

Die Variable enthält die Position des Rückzugspunktes (XYZ-Koordinaten des Parameters JSAFE).

Eingabeformat

Bsp.: <Positionsvariable> = P_SAFE [(<Mechanismusnummer>)]

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Roboternummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1  P1 = P_Safe                    'Überträgt die Daten des Rückzugspunktes in die
                                  Positionsvariable P1
```

Erläuterung

- Die Variable überträgt die aus den Gelenkdaten des Parameters JSAFE ermittelten XYZ-Koordinaten des Rückzugspunktes.
- Die Variable kann ausschließlich gelesen werden.

7.2.60 P_Tool/P_NTool

Funktion: Werkzeug-Konvertierungsdaten lesen

Die Variable enthält die Werte der Werkzeug-Konvertierungsdaten.

P_Tool: aktuell eingestellte Werkzeug-Konvertierungsdaten

P_NTool: Standardwert der Werkzeug-Konvertierungsdaten (0, 0, 0, 0, 0, 0, 0, 0) (0, 0)

Eingabeformat

Bsp.: <Positionsvariable> = P_Tool [(<Mechanismusnummer>)]

Bsp.: <Positionsvariable> = P_NTool

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest 1 ≤ Mechanismusnummer ≤ 3 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
1 P1 = P_Tool          'Überträgt die aktuell eingestellten Werkzeug-
                       Konvertierungsdaten in die Positionsvariable P1
2 Base P_NTool         'Setzt die Basis-Konvertierungsdaten auf den
                       Standardwert zurück
```

Erläuterung

- Die Variable P_Tool enthält die über den Tool-Befehl oder den Parameter MEXTL festgelegten Werkzeug-Konvertierungsdaten.
- Eine Änderung der Werkzeugdaten erfolgt über den Tool-Befehl.
- Die Variablen können ausschließlich gelesen werden.

7.2.61 P_WkCord

Funktion: Werkstückkoordinaten lesen

Die Variable ermöglicht einen Zugriff auf die Werkstückkoordinaten, die zuletzt festgelegt wurden, und sie erlaubt die Einstellung neuer Koordinaten. Die zugehörigen Parameter sind die Parameter WK1CORD bis WK8CORD.

Eingabeformat

Bsp.: <Positionsvariable> = P_WkCord [(<Werkstückkoordinatennummer>)] <- Zugriff
Bsp.: P_WkCord [(<Werkstückkoordinatennummer>)] = <Werkstückkoordinatendaten> <- Einstellung

<Positionsvariable>	Legt eine Positionsvariable fest
<Werkstückkoordinatennummer>	Legt die Werkstückkoordinatennummer fest 1 ≤ Werkstückkoordinatennummer ≤ 8 Es können Konstanten, Variablen, logische/arithmetische Ausdrücke und Funktionen verwendet werden. Bei Zahlen vom Typ Real oder vom Typ Real mit doppelter Genauigkeit wird eine Nachkommastelle, die größer oder gleich 5 ist, aufgerundet, ansonsten abgerundet.
<Werkstückkoordinatendaten>	Legt die Werkstückkoordinatendaten über eine Positionskonstante oder eine Positionsvariable fest Die festgelegten Werte (Koordinaten) beschreiben die Position des Nullpunkt des Werkstückkoordinatensystems vom Basiskoordinatensystem aus gesehen.

Programmbeispiel

```
1 PW = P_WkCord (1)          'Liest die Werkstückkoordinaten 1 (eingestellter Wert
                             des Parameters WK1CORD) ein und weist sie der
                             Positionsvariable PW zu
2 PW.X = PW.X + 100          'Addiert 100 zu der X-Koordinate des eingelesenen
                             Wertes
3 PW.Y = PW.Y + 100          'Addiert 100 zu der Y-Koordinate des eingelesenen
                             Wertes
4 P_WkCord (2) = PW          'Überträgt das Ergebniss der obigen Operationen in die
                             Werkstückkoordinaten 2 (eingestellter Wert des
                             Parameters WK2CORD)
5 Base 2                     'Definiert die Werkstückkoordinaten 2 als Ursprung
                             des neuen Weltkoordinatensystems
6 Mov P1                     'Fährt die Position P1 im neuen Koordinatensystem
                             mit Gelenk-Interpolation an
```

Erläuterung

- Durch die Festlegung einer Werkstückkoordinatenummer können die entsprechenden Werkstückkoordinatenwerte gelesen oder vorgegeben werden. Die Werkstückkoordinatennummern 1 bis 8 entsprechen den Parametern WK1CORD bis WK8CORD.
- Die X-, Y- und Z-Koordinaten der Werkstückkoordinatendaten geben die parallele Verschiebung des Werkstückkoordinatensystems in Bezug auf das Basiskoordinatensystem an. Die Komponenten A, B und C geben dabei die Drehwinkel des Werkstückkoordinatensystems in Bezug auf das Weltkoordinatensystem an.
  X = Paralleler Abstand zur X-Achse
  Y = Paralleler Abstand zur Y-Achse
  Z = Paralleler Abstand zur Z-Achse
  A = Drehwinkel um die X-Achse
  B = Drehwinkel um die Y-Achse
  C = Drehwinkel um die Z-Achse
  L1 = Weg der Zusatzachse 1
  L2 = Weg der Zusatzachse 2
- Aus Sicht des Koordinatenursprungs werden Winkel in Uhrzeigerrichtung positiv gewertet.
- Die Werte der Stellungsmerker sind ohne Bedeutung.
- Eine Festlegung der Werkstückkoordinaten über diesen Parameter löscht die Daten WO, WX und WY der entsprechenden Werkstückkoordinatennummer (Koordinatenwerte von 3 Punkten, die als Werkstückkoordinaten geteacht wurden. Parameter: WKnWO, WKnWX und WKnWY (n: 1–8)).

Beispiel ▽

Durch die Ausführung des Programmschritts 4 (P_WkCORD (2) = PW) im Programmbeispiel oben werden die Parameter WK2WO, WK2WX und WK2WY auf „0“ gesetzt.

△

Steht in Beziehung zu folgenden Befehlen:

Base

Steht in Beziehung zu folgenden Parametern:

MEXBSNO, WKnCORD (n = 1 bis 8), WKnWO, WKnWX, WKnWY (n = 1 bis 8)

7.2.62 P_Zero

Funktion: Initialisierungswerte lesen

Die Variable enthält die Daten (0, 0, 0, 0, 0, 0, 0, 0) (0, 0).

Eingabeformat

Bsp.: <Positionsvariable> = P_Zero

<Positionsvariable> Legt eine Positionsvariable fest

Programmbeispiel

```
1  P1 = P_Zero                    'Überträgt die Daten (0, 0, 0, 0, 0, 0, 0, 0) (0, 0)
                                  in die Positionsvariable P1
```

Erläuterung

- Die Variable dient zur Initialisierung von Positionsvariablen.
- Die Variable kann ausschließlich gelesen werden.

8 Funktionen

8.1 Allgemeine Hinweise

In den nachfolgenden Abschnitten finden Sie eine alphabetische Auflistung aller Funktionen und deren Anwendungsmöglichkeiten.

8.1.1 Beschreibung des verwendeten Formats

Funktion

Hier finden Sie eine Beschreibung der Funktion.

Eingabeformat

Hier finden Sie das genaue Format zur Eingabe der Funktion.

Programmbeispiel

Hier finden Sie die Verwendung der Funktion in einem Beispielprogramm.

Erläuterung

Hier finden Sie eine detaillierte Beschreibung, Besonderheiten usw. der Funktion.

8.2 Detaillierte Funktionsbeschreibung

In diesem Abschnitt finden Sie eine detaillierte Beschreibung sowie Programmbeispiele zur Anwendung der Funktionen.

8.2.1 Abs

Funktion: Betrag berechnen

Die Funktion bildet den Betrag des angegebenen Wertes.

Eingabeformat

<Numerische Variable> = ABS (<Numerischer Ausdruck>)

Programmbeispiel

```
1 P2.C = Abs(P1.C)      'Überträgt die C-Komponente von P1 ohne Vorzeichen
                        in die C-Komponente der Positionsvariablen P2
2 Mov P2                'Position P2 mittels Gelenk-Interpolation anfahren
3 M2 = 100              'Weist der Variablen M2 den Wert „100“ zu
4 M1 = Abs(M2)          'Weist der Variablen M1 den Wert „100“ zu
```

Erläuterung

- Die Funktion ABS bildet den Betrag des angegebenen Wertes, so dass das Ergebnis ein positives Vorzeichen hat.

Steht in Beziehung zu folgenden Funktionen:

Sgn

8.2.2 Align

Funktion: axiale Ausrichtung

Setzt den Wert der Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand zur Stellung (A, B, C) der aktuellen Position. Das Ergebnis der Align-Funktion ist ein numerischer Wert. Die Funktion beinhaltet auch Bewegungsbefehle wie z. B. den Mov-Befehl.

Eingabeformat

<Positionsvariable> = Align (<Position>)

Programmbeispiel

```
1 P1 = P_Curr          'Überträgt die Daten der aktuellen Position in die
                       Positionsvariable P1
2 P2 = Align(P1)       'Überträgt die Position mit dem kleinstmöglichen
                       senkrechten oder waagerechten Abstand zur Stellung
                       der aktuellen Position in P2
3 Mov P2               'Richtet den Roboterarm axial aus
```

Erläuterung

- Die Funktion ALIGN setzt den Wert der Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand (0°, ±90°, ±180°) zur Stellung (A, B,C) der aktuellen Position.
- Da das Operationsergebnis eine Positionsvariable ist, erfolgt eine Fehlermeldung, wenn auf der linken Seite der Gleichung eine Gelenkvariable angegeben wird.
- Die Funktion kann nicht bei 5-achsigen Robotern verwendet werden, da die Orientierungsdaten A, B und C eine andere Bedeutung haben.

Beispiel ▽

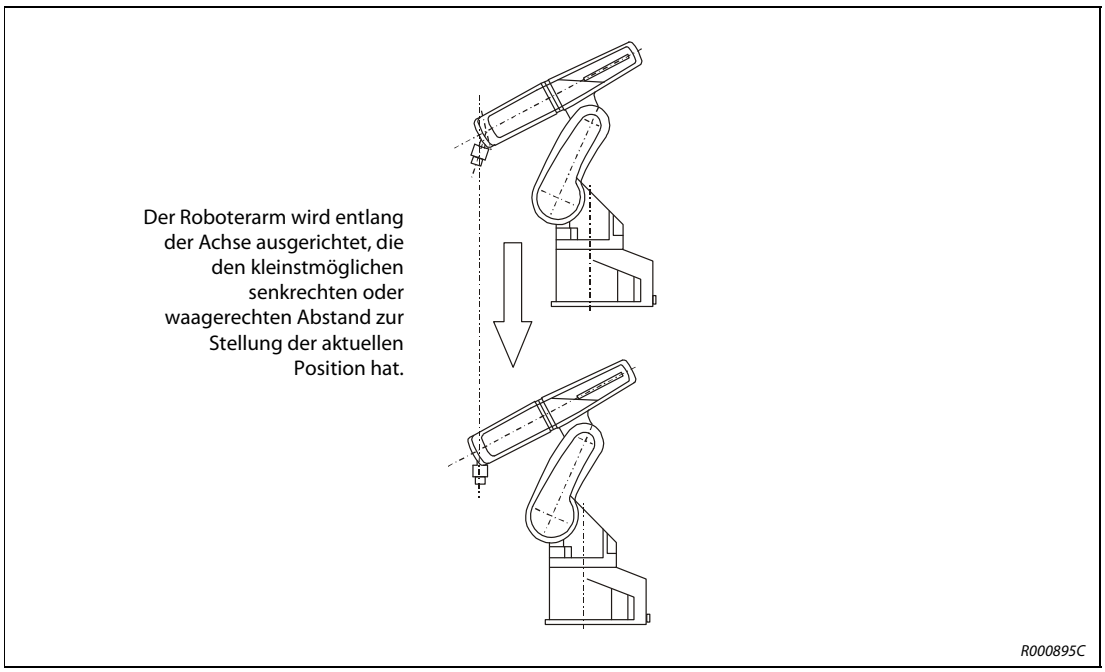

Der Roboterarm wird entlang der Achse ausgerichtet, die den kleinstmöglichen senkrechten oder waagerechten Abstand zur Stellung der aktuellen Position hat.
R000895C

Abb. 8-1: *Axiale Ausrichtung des Roboterarms*

△

8.2.3 Asc

Funktion: ASCII-Code erzeugen

Die Funktion erzeugt den ASCII-Code für das erste Zeichen in der Zeichenkette.

Eingabeformat

<Numerische Variable> = Asc (<Zeichenkette>)

Programmbeispiel

```
1  M1 = Asc("A")                    'Weist der numerischen Variablen M1 den Wert
                                    „&H41“ zu
```

Erläuterung

- Die Funktion Asc bildet den ASCII-Code des ersten Zeichens einer Zeichenkette.
- Bei Anwendung der Asc-Funktion auf eine Leerkette erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Chr$, Val, Cvi, Cvs, Cvd

8.2.4 Atn/Atn2

Funktion: Arcus Tangens berechnen

Die Funktion berechnet den Arcus Tangens.

Eingabeformat

<Numerische Variable> = Atn (<Numerischer Ausdruck>)

<Numerische Variable> = Atn2 (<Numerischer Ausdruck 1>, <Numerischer Ausdruck 2>)

<Numerische Variable>	Legt eine numerische Variable zur Übertragung des berechneten Wertes im Bogenmaß (Radiant) fest
<Numerischer Ausdruck>	Quotient $\Delta Y/\Delta X$
<Numerischer Ausdruck 1>	ΔY
<Numerischer Ausdruck 2>	ΔX

Programmbeispiel

```
1 M1 = Atn(100/100)          'Weist der numerischen Variablen M1 den Wert des
                             Arcus Tangens von π/4 in Radiant zu
2 M2 = Atn2(100,100)         'Weist der numerischen Variablen M2 den Wert
                             des Arcus Tangens von –π/4 in Radiant zu
```

Erläuterung

- Die Funktionen berechnen den Arcus Tangens. Die Einheit ist Radiant.
- Der Wertebereich der Funktion ATN ist $-\pi/2$ bis $\pi/2$.
- Der Wertebereich der Funktion ATN2 ist $-\pi$ bis π.
- Geht <Numerischer Ausdruck 2> gegen 0, ergibt der Atn2 bei einem positiven <Numerischen Ausdruck 1> $\pi/2$ und bei einem negativen <Numerischen Ausdruck 1> $-\pi/2$.
- Auf die Argumente <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion Atn2 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

  Nicht erlaubt sind z. B.: M1 = Atn2(Max(MA,MB),100)
  M1 = Atn2(Cint(10.2),100)

Steht in Beziehung zu folgenden Funktionen:

Sin, Cos, Tan

8.2.5 Bin$

Funktion: binäre Zeichenkette erzeugen

Die Funktion wandelt einen Wert in eine binäre Zeichenkette um.

Eingabeformat

<Zeichenkettenvariable> = Bin$ (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = &B11111111          'Weist der numerischen Variablen M1 den Wert der
                           binären Zeichenkette zu
2 C1$ = Bin$(M1)           'Weist der Zeichenkettenvariablen die Zeichenkette
                           „11111111“ zu
```

Erläuterung

- Die Funktion Bin$ konvertiert den angegebenen Wert in eine binäre Zeichenkette.
- Ist der numerische Ausdruck keine Integer-Zahl, wird der Wert gerundet und anschließend in eine binäre Zeichenkette umgewandelt.
- Die Umkehrfunktion von Bin$ ist Val.

Steht in Beziehung zu folgenden Funktionen:

Hex$, Str$, Val

8.2.6 CalArc

Funktion: Kreisbogen berechnen

Die Funktion berechnet den über drei Punkte festgelegten Kreisbogen.

Eingabeformat

```
<Numerische Variable 4> = CalArc (<Position 1>, <Position 2>, <Position 3>,
                                  <Numerische Variable 1>, <Numerische Variable 2>,
                                  <Numerische Variable 3>, <Positionsvariable 1>)
```

<Position 1>	Legt den Startpunkt des Kreisbogens fest
<Position 2>	Legt die Zwischenposition auf dem Kreisbogen fest
<Position 3>	Legt den Endpunkt des Kreisbogens fest (Die drei Punkte entsprechen denen im Mvr-Befehl.)
<Numerische Variable 1>	Berechnung des Radius des festgelegten Kreisbogens in mm
<Numerische Variable 2>	Berechnung des Zentriwinkels des festgelegten Kreisbogens in Radiant
<Numerische Variable 3>	Berechnung der Kreisbogenlänge des festgelegten Kreisbogens in mm
<Positionsvariable 1>	Berechnung der Mittelpunktkoordinaten des festgelegten Kreisbogens in mm (Die Daten werden als Positionsdaten übertragen. A, B, C sind gleich „0“.)
<Numerische Variable 4>	Übertragener Wert: 1: Berechnung ausgeführt –1: Zwei der Positionen 1 bis 3 sind deckungsgleich oder alle drei Positionen liegen auf einer Geraden. –2: Alle drei Punkte sind annähernd deckungsgleich.

Programmbeispiel

```
1  M1 = CalArc(P1,P2,P3,M10,M20,M30,P10)  'Berechnung des Kreisbogens
2  If M1 <> 1 Then End                    'Programm bei Fehler in der Berechnung des
                                          Kreisbogens beenden
3  MR = M10                               'Überträgt den Radius in die numerische Variable MR
4  MRD = M20                              'Überträgt den Zentriwinkel in die numerische
                                          Variable MRD
5  MARCLEN = M30                          'Überträgt die Bogenlänge in die numerische
                                          Variable MARCLEN
6  PC = P10                               'Überträgt die Position des Mittelpunktes in die
                                          Positionsvariable PC
```

Erläuterung

- Die Funktion berechnet die Daten des über die Positionen 1, 2 und 3 festgelegten Kreisbogens.
- Bei erfolgreicher Berechnung des Kreisbogens wird die numerische Variable 4 auf „1“ gesetzt.
- Sind 2 der drei Positionen deckungsgleich oder liegen alle drei Positionen auf einer Geraden, wird die numerische Variable 4 auf „–1“ gesetzt. In diesem Fall wird die Entfernung zwischen Start- und Endpunkt als Bogenlänge, „–1“ als Radius, „0“ als Zentriwinkel und (0, 0, 0) als Mittelpunktposition übertragen.
- Tritt bei der Berechnung des Kreisbogens ein Fehler auf, wird die numerische Variable 4 auf „–2“ gesetzt. In diesem Fall wird „0“ als Bogenlänge, „–1“ als Radius, „0“ als Zentriwinkel und (0, 0, 0) als Mittelpunktposition übertragen.
- Auf die Argumente <Position 1>, <Position 2>, <Position 3>, <Numerische Variable 1>, <Numerische Variable 2>, <Numerische Variable 3> und <Positionsvariable 1> der Funktion CalArc darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.7 Chr$

Funktion: Zeichen erzeugen

Die Funktion erzeugt ein Zeichen entsprechend dem angegebenen Wert.

Eingabeformat

<Zeichenkettenvariable> = Chr$ (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = &H40                 'Weist der numerischen Variablen M1 den Wert
                            „&H40“ zu
2 C1$ = Chr$(M1+1)          'Weist der Zeichenkettenvariablen C1$ die Zeichen-
                            kette „A“ zu
```

Erläuterung

- Die Funktion Chr$ erzeugt das Zeichen, das dem angegebenen numerischen Wert entspricht.
- Ist der numerische Ausdruck keine Integer-Zahl, wird der Wert gerundet und anschließend in das entsprechende Zeichen umgewandelt.

Steht in Beziehung zu folgenden Funktionen:

Asc

8.2.8 Cint

Funktion: Integer-Zahl berechnen

Die Funktion berechnet eine Integer-Zahl.

Eingabeformat

<Numerische Variable> = Cint (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Cint(1.5)            'Weist der numerischen Variablen M1 den Wert „2“ zu
2 M2 = Cint(1.4)            'Weist der numerischen Variablen M2 den Wert „1“ zu
3 M3 = Cint(–1.4)           'Weist der numerischen Variablen M3 den Wert „–1“ zu
4 M4 = Cint(–1.5)           'Weist der numerischen Variablen M4 den Wert „–2“ zu
```

Erläuterung

- Die Funktion Cint rundet den Wert eines numerischen Ausdrucks zu einer Integer-Zahl.

Steht in Beziehung zu folgenden Funktionen:

Int, Fix

8.2.9 CkSum

Funktion: Prüfsumme erzeugen

Die Funktion erzeugt die Prüfsumme einer Zeichenkette.

Eingabeformat

<Numerische Variable> = CkSum (<Zeichenkette>, <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2>)

<Zeichenkette>	Legt die Zeichenkette fest, aus der die Prüfsumme gebildet werden soll
<Numerischer Ausdruck 1>	Legt die Position des ersten Zeichens fest, ab dem die Prüfsummenbildung startet
<Numerischer Ausdruck 1>	Legt die Position des ersten Zeichens fest, bei dem die Prüfsummenbildung endet

Programmbeispiel

```
1  M1 = CkSum("ABCDEFG",1,3)        'Weist der numerischen Variablen M1 den Wert
                                    &H41 ("A") + &H42 ("B") + &H43 ("C") = &HC6 zu
```

Erläuterung

- Die Funktion CkSum addiert die Werte einer Zeichenkette vom angegebenen Startpunkt bis zum angegebenen Endpunkt und erzeugt so die Prüfsumme in einem Wertebereich zwischen 0 bis 255.
- Liegt der angegebene Startpunkt außerhalb der Zeichenkette, erfolgt eine Fehlermeldung.
- Liegt der angegebene Endpunkt außerhalb der Zeichenkette, erfolgt die Prüfsummenbildung vom angegebenen Startzeichen bis zum letzten Zeichen der Zeichenkette.
- Ist das Ergebnis größer als 255, wird der Wert modifiziert.
- Auf die Argumente <Zeichenkette>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion CkSum darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.10 Cos

Funktion: Cosinus berechnen

Die Funktion berechnet den Cosinus.

Eingabeformat

<Numerische Variable> = Cos (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Cos(Rad(60))          'Weist der numerischen Variablen M1 den Wert „0.5“ zu
```

Erläuterung

- Die Funktion berechnet den Cosinus des numerischen Ausdrucks. Die Einheit des Arguments ist Radiant.
- Als Definitionsbereich ist der gesamte gültige Zahlenbereich zugelassen.
- Der Wertebereich der Funktion Cos ist –1 bis 1.

Steht in Beziehung zu folgenden Funktionen:

Sin, Tan, Atn/Atn2

8.2.11 Cvi

Funktion: 2 Zeichen einer Zeichenkette umwandeln

Die Funktion berechnet die Integer-Werte der ersten beiden Zeichen einer Zeichenkette.

Eingabeformat

<Numerische Variable> = Cvi (<Zeichenkette>)

Programmbeispiel

```
1 M1 = Cvi("10ABC")          'Weist der numerischen Variablen M1 den Wert
                             „&H3031“ zu
```

Erläuterung

- Die Funktion Cvi wandelt die ersten beiden Zeichen einer Zeichenkette in Integer-Werte um.
- Besteht die Zeichenkette aus einem oder weniger Zeichen, erfolgt eine Fehlermeldung.
- Die Umwandlung eines numerischen Ausdrucks in eine Zeichenkette mit zwei Zeichen erfolgt mit der Funktion Mki$.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

Asc, Cvs, Cvd, Mki$, Mks$, Mkd$

8.2.12 Cvs

Funktion: 4 Zeichen einer Zeichenkette umwandeln

Die Funktion berechnet den Real-Wert mit einfacher Genauigkeit der ersten 4 Zeichen einer Zeichenkette.

Eingabeformat

<Numerische Variable> = Cvs (<Zeichenkette>)

Programmbeispiel

```
1 M1 = Cvs("FFFF")          'Weist der numerischen Variablen M1 den Wert
                            „12689,6“ zu
```

Erläuterung

- Die Funktion Cvs wandelt die ersten 4 Zeichen einer Zeichenkette in einen Real-Wert mit einfacher Genauigkeit um.
- Besteht die Zeichenkette aus drei oder weniger Zeichen, erfolgt eine Fehlermeldung.
- Die Umwandlung eines numerischen Ausdrucks in eine Zeichenkette mit 4 Zeichen erfolgt mit der Funktion Mks$.

Steht in Beziehung zu folgenden Funktionen:

Asc, Cvi, Cvd, Mki$, Mks$, Mkd$

8.2.13 Cvd

Funktion: 8 Zeichen einer Zeichenkette umwandeln

Die Funktion berechnet den Real-Wert mit doppelter Genauigkeit der ersten 8 Zeichen einer Zeichenkette.

Eingabeformat

<Numerische Variable> = Cvd (<Zeichenkette>)

Programmbeispiel

```
1 M1 = Cvd("FFFFFFFF")      'Weist der numerischen Variablen M1 den Wert
                            „+3,52954E+30“ zu
```

Erläuterung

- Die Funktion Cvd wandelt die ersten 8 Zeichen einer Zeichenkette in einen Real-Wert mit doppelter Genauigkeit um.
- Besteht die Zeichenkette aus sieben oder weniger Zeichen, erfolgt eine Fehlermeldung.
- Die Umwandlung eines numerischen Ausdrucks in eine Zeichenkette mit 8 Zeichen erfolgt mit der Funktion Mkd$.

Steht in Beziehung zu folgenden Funktionen:

Asc, Cvi, Cvs, Mki$, Mks$, Mkd$

8.2.14 Deg

Funktion: Radiant in Grad umwandeln

Die Funktion wandelt einen in Radiant angegebenen Winkel in Grad um.

Eingabeformat

<Numerische Variable> = Deg (<Numerischer Ausdruck>)

Programmbeispiel

```
1 P1 = P_Curr                                  'Weist der Positionsvariablen P1den
                                               Wert der aktuellen Position zu
2 If Deg(P1.C) < 170 OR Deg(P1.C) > –150 Then *NOERR   'Sprung zur Marke NOERR, falls die
                                               C-Komponente der Position P1
                                               innerhalb des Bereichs von –150° bis
                                               170° liegt
3 Error 9100                                   'Generiert den Fehler mit der Fehler-
                                               nummer 9100
4 *NOERR                                       'Definiert die Marke NOERR
```

Erläuterung

- Die Funktion Deg wandelt einen in Radiant (rad) angegebenen Winkel in einen Winkel mit der Einheit Grad (deg) um.
- Bei einem Zugriff auf die Winkel von Positionsdaten über Positionskonstanten, erfolgt die Anzeige der Winkel in Grad (500, 0, 600, 180, 0, 180) (7, 0). Bei einem direkten Zugriff auf die Winkel über die Angabe der Komponente (z. B. über P1.C) ist die Einheit Radiant. Eine Umstellung der Einheit auf Grad kann durch Einstellung des Parameters PRGMDEG auf „1“ erfolgen.

Steht in Beziehung zu folgenden Funktionen:

Rad

8.2.15 Dist

Funktion: Abstand berechnen

Die Funktion berechnet den Abstand zwischen zwei Punkten (Positionsvariablen).

Eingabeformat

<Numerische Variable> = Dist (<Position 1>, <Position 2>)

Programmbeispiel

```
1  M1 = Dist(P1,P2)        'Weist der numerischen Variablen M1 den Wert des
                            Abstands zwischen den Positionen P1 und P2 zu
```

Erläuterung

- Die Funktion Dist berechnet den Abstand zwischen zwei Positionen in mm.
- Für die Berechnung werden nur die XYZ-Koordinaten verwendet. Die Stellungsdaten sind ohne Bedeutung.
- Gelenk-Variablen können zur Berechnung nicht verwendet werden. Wird die Funktion mit Gelenk-Variablen verwendet, erfolgt bei der Ausführung eine Fehlermeldung.
- Auf die Argumente <Position 1> und <Position 2> der Funktion Dist darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.16 Exp

Funktion: Exponentialfunktion berechnen

Die Funktion berechnet die Exponentialfunktion mit der Basis „e".

Eingabeformat

<Numerische Variable> = Exp (<Numerischer Ausdruck>)

Programmbeispiel

```
1  M1 = Exp(2)             'Weist der numerischen Variablen M1 den Wert „e2" zu
```

Erläuterung

- Die Funktion Exp berechnet den Wert der Exponentialfunktion bei einem numerischen Ausdruck.

Steht in Beziehung zu folgenden Funktionen:

Ln

8.2.17 Fix

Funktion: Integer-Anteil bilden

Die Funktion berechnet den ganzzahligen Anteil eines numerischen Ausdrucks.

Eingabeformat

<Numerische Variable> = Fix (<Numerischer Ausdruck>)

Programmbeispiel

```
1  M1 = Fix(5.5)                    'Weist der numerischen Variablen M1 den Wert „5“ zu
```

Erläuterung

- Die Funktion Fix berechnet den Integer-Anteil eines numerischen Ausdrucks.
- Bei einem positiven numerischen Ausdruck liefert die Funktion dasselbe Ergebnis wie die Funktion Int.
- Bei einem negativen numerischen Ausdruck werden die Stellen hinter dem Komma folgendermaßen gerundet: Fix(–2.3) = 2.0.

Steht in Beziehung zu folgenden Funktionen:

Cint, Int

8.2.18 Fram

Funktion: Koordinatensystem berechnen

Die Funktion berechnet ein Koordinatensystem (Fläche) über drei Punkte. Verwenden Sie zur Berechnung einer Palette die Befehle Def Plt und Plt.

Eingabeformat

<Position 4> = Fram (<Position 1>,<Position 2>,<Position 3>)

<Position 1>	Legt die XYZ-Koordinaten des Flächenursprungs der über drei Positionen definierten Fläche als Variable oder Konstante fest
<Position 2>	Legt einen Punkt auf der X-Achse in der über drei Positionen definierten Fläche als Variable oder Konstante fest
<Position 3>	Legt einen Punkt in positiver Richtung auf der Y-Achse in der über drei Positionen definierten Fläche als Variable oder Konstante fest
<Position 4>	Positionsvariable zur Ablage des Ergebnisses Der Stellungsmerker entspricht ab Software-Version J1 dem der <Position 1>

Programmbeispiel

```
1 Base P_NBase                  'Setzt die Basis-Konvertierungsdaten auf den Standardwert zurück
2 P100 = Fram(P1,P2,P3)         'Erzeugt ein über die Positionen P1, P2 und P3 definiertes Koordinatensystem
3 P10 = Inv(P10)                'Weist der Positionsvariablen P10 die inverse Matrix von P10 zu
4 Base P10                      'Mit Hilfe von P10 wird ein neues Weltkoordinatensystem definiert
  :
```

Erläuterung

- Die Funktion Fram kann zur Definition des Basis-Koordinatensystems verwendet werden.
- Die Funktion definiert den Ursprung und die Neigung einer Fläche über die XYZ-Koordinaten der drei angegebenen Positionen. Das Ergebnis wird in eine Positionsvariable übertragen. Die XYZ-Koordinaten der Position entsprechen denen der Position 1. Die Stellungsdaten A, B, und C geben die über die drei Positionen festgelegte Neigung der Fläche wieder.
- Da bei Ausführung der Funktion Positionsdaten übertragen werden, darf als Variable 4 keine Gelenkvariable verwendet werden. Bei Verwendung einer Gelenkvariablen erfolgt eine Fehlermeldung.
- Auf die Argumente <Position 1>, <Position 2> und <Position 3> der Funktion Fram darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

  Nicht erlaubt ist z. B.: P10 = Fram(FPrm(P01,P02,P03), P04, P05)

- Weitere Hinweise zu dieser Funktion finden Sie im Abschn. 9.9.

8.2.19 Hex$

Funktion: numerischen Ausdruck in hexadezimale Zeichenkette umwandeln

Die Funktion wandelt den Wert eines numerischen Ausdrucks (zwischen –32768 und 32767) in eine hexadezimale Zeichenkette um.

Eingabeformat

<Zeichenkettenvariable> = Hex$ (<Numerischer Ausdruck> [,<Anzahl der Zeichen>])

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Numerischer Ausdruck>	Legt den umzuwandelnden numerischen Ausdruck fest
<Anzahl der Zeichen>	Legt die Anzahl der auszugebenden Zeichen fest

Programmbeispiel

```
1 C1$ = Hex$(&H41FF)          'Weist der Zeichenkettenvariablen C1$ die
                              Zeichenkette „41FF“ zu
2 C2$ = Hex$(&H41FF,2)        'Weist der Zeichenkettenvariablen C2$ die
                              Zeichenkette „FF“ zu
```

Erläuterung

- Bei Angabe der <Anzahl der Zeichen> wird die festgelegte Zeichenkettenlänge, beginnend mit dem äußersten rechten Zeichen, ausgegegeben.
- Ist der numerische Ausdruck keine Integer-Zahl, wird der Wert gerundet und anschließend in eine hexadezimale Zeichenkette umgewandelt.
- Die Umkehrung der Funktion Hex$ erfolgt über die Funktion Val.
- Auf das Argument <Anzahl der Zeichen> der Funktion Hex$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

  Nicht erlaubt ist z. B.: C1$ = Hex$(ASC("a"),1)

Steht in Beziehung zu folgenden Funktionen:

Bin$, Str$, Val

8.2.20 Int

Funktion: Integer-Zahl erzeugen

Die Funktion erzeugt die größtmögliche Integer-Zahl, die kleiner als der Wert des numerischen Ausdrucks ist.

Eingabeformat

<Numerische Variable> = Int (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Int(3.3)          'Weist der numerischen Variablen M1 den Wert „3“ zu
```

Erläuterung

- Bei einem positiven numerischen Ausdruck liefert die Funktion dasselbe Ergebnis wie die Funktion Fix.
- Bei einem negativen numerischen Ausdruck werden die Stellen hinter dem Komma folgendermaßen gerundet: Int(–2.3) = 3.0.

Steht in Beziehung zu folgenden Funktionen:

Cint, Fix

8.2.21 Inv

Funktion: Position invertieren

Die Funktion erzeugt die inverse Matrix der angegebenen Position.

Eingabeformat

<Positionsvariable> = Inv (<Positionsvariable>)

Programmbeispiel

```
1 P1 = Inv(P2)           'Weist der Positionsvariablen P1 die inverse Matrix von
                         P2 zu
```

Erläuterung

- Die Funktion Inv wird bei relativen Rechenoperationen mit Positionen verwendet.
- Als Argument dürfen keine Gelenkvariablen verwendet werden. Bei Verwendung von Gelenkvariablen erfolgt eine Fehlermeldung.
- Da bei Ausführung der Funktion Positionsdaten übertragen werden, darf auf der linken Seite der Gleichung keine Gelenkvariable verwendet werden. Bei Verwendung einer Gelenkvariablen erfolgt eine Fehlermeldung.

8.2.22 JtoP

Funktion: Gelenk- in Positionsdaten umwandeln

Die Funktion wandelt die angegebenen Gelenkdaten in Positionsdaten um.

Eingabeformat

<Positionsvariable> = JtoP (<Gelenkvariable>)

Programmbeispiel

```
1  P1 = JtoP(J1)          'Weist die über die Gelenkvariable J1 festgelegte
                          Position der Positionsvariablen P1 im XYZ-Format zu
```

Erläuterung

- Als Argument dürfen keine Positionsvariablen verwendet werden. Bei Verwendung von Positionsvariablen erfolgt eine Fehlermeldung.
- Da bei Ausführung der Funktion Positionsdaten übertragen werden, darf auf der linken Seite der Gleichung keine Gelenkvariable verwendet werden. Bei Verwendung einer Gelenkvariablen erfolgt eine Fehlermeldung.
- Bei der Software-Version P8 (Serie SD) bzw. N8 (Serie SQ) ist der Mechanismus standardmäßig mit der Nummer „1“ definiert. Wenn nach der Ausführung des RelM-Befehls dem Mechanismus die Nummer „1“ zugeordnet wurde oder wenn die Programmplatznummer (Slot) ungleich „1“ ist, dann ist die Ausführung des GetM-Befehls nicht notwendig. Ist die Mechanismusnummer nicht „1“, muss zuvor der GetM-Befehl ausgeführt werden.

Steht in Beziehung zu folgenden Funktionen:

PtoJ

8.2.23 Left$

Funktion: Teil einer Zeichenkette erzeugen

Die Funktion erzeugt einen Teil der angegebenen Zeichenkette, beginnend mit dem linken Zeichen.

Eingabeformat

<Zeichenkettenvariable> = Left$ (<Zeichenkette>, <Numerischer Ausdruck>)

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Numerischer Ausdruck>	Legt die Anzahl der auszugebenden Zeichen fest

Programmbeispiel

```
1 C1$ = Left$("ABC",2)          'Weist der Zeichenkettenvariablen C1$ die
                                 Zeichenkette „AB“ zu
```

Erläuterung

- Die Funktion erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, beginnend mit dem linken Zeichen, ist im zweiten Argument festgelegt.
- Ist die Anzahl der auszugebenden Zeichen negativ oder größer als die Länge der Zeichenkette, erfolgt eine Fehlermeldung.
- Auf die Argumente <Zeichenkette> und <Numerischer Ausdruck> der Funktion Left$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Mid$, Right$

8.2.24 Len

Funktion: Länge einer Zeichenkette berechnen

Die Funktion berechnet die Länge einer Zeichenkette.

Eingabeformat

<Numerische Variable> = Len (<Zeichenkette>)

Programmbeispiel

```
1 M1 = Len("ABCDEFG")           'Weist der numerischen Variablen M1 den Wert „7“ zu
```

Erläuterung

- Die Funktion Len berechnet die Länge der angebene Zeichenkette.

Steht in Beziehung zu folgenden Funktionen:

Left$, Mid$, Right$

8.2.25 Ln

Funktion: natürlichen Logarithmus berechnen

Die Funktion berechnet den natürlichen Logarithmus (Basis: e).

Eingabeformat

<Numerische Variable> = Ln (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Ln(2)                    'Weist der numerischen Variablen M1 den Wert
                                „0,693147“ zu
```

Erläuterung

- Die Funktion Ln berechnet den natürlichen Logarithmus des angegebenen numerischen Ausdrucks.
- Ist der numerische Ausdruck Null oder negativ, erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Exp, Log

8.2.26 Log

Funktion: dekadischen Logarithmus berechnen

Die Funktion berechnet den dekadischen Logarithmus (Basis: 10).

Eingabeformat

<Numerische Variable> = Log (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Log(2)                   'Weist der numerischen Variablen M1 den Wert
                                „0,301030“ zu
```

Erläuterung

- Die Funktion Log berechnet den dekadischen Logarithmus des angegebenen numerischen Ausdrucks.
- Ist der numerische Ausdruck Null oder negativ, erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Exp, Ln

8.2.27 Max

Funktion: Maximalwert berechnen

Die Funktion berechnet den Maximalwert.

Eingabeformat

<Numerische Variable> = Max (<Numerischer Ausdruck 1>, <Numerischer Ausdruck 2>, ...)

Programmbeispiel

```
1  M1 = Max(2,1,3,4,10,100)        'Weist der numerischen Variablen M1 den Wert „100“ zu
```

Erläuterung

- Die Funktion Max berechnet den maximalen Wert der angegebenen numerischen Ausdrücke.
- Die maximale Zeilenlänge wird durch die in einer Zeile zulässige Anzahl von Zeichen begrenzt (240 Zeichen).
- Auf die Argumente <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2> und <...> der Funktion Max darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Min

8.2.28 Mid$

Funktion: Teil einer Zeichenkette erzeugen

Die Funktion erzeugt, beginnend mit der festgelegten Position, einen Teil einer Zeichenkette.

Eingabeformat

```
<Zeichenkettenvariable> = Mid$ (<Zeichenkette>, <Numerischer Ausdruck 1>,
                                <Numerischer Ausdruck 2>])
```

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Numerischer Ausdruck 1>	Legt die Position des ersten Zeichens fest
<Numerischer Ausdruck 2>	Legt die Länge der Zeichenkette fest

Programmbeispiel

```
1  C1$ = Mid$("ABCDEFG",3,2)        'Weist der Zeichenkettenvariablen C1$ die
                                    Zeichenkette „CD“ zu
```

Erläuterung

- Die Funktion erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, ist im numerischen Ausdruck 2, die Position des ersten Zeichens im numerischen Ausdruck 1 festgelegt.
- Ist die Angabe für die Anzahl der auszugebenden Zeichen oder die Position des ersten Zeichens Null oder negativ, erfolgt eine Fehlermeldung. Liegt die Position eines der zu erfassenden Zeichen außerhalb der Zeichenkette, erfolgt eine Fehlermeldung.
- Auf die Argumente <Zeichenkette>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion Mid$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Left$, Right$, Len

8.2.29 Min

Funktion: Minimalwert berechnen

Die Funktion berechnet den Minimalwert.

Eingabeformat

<Numerische Variable> = Min (<Numerischer Ausdruck 1>, <Numerischer Ausdruck 2>, ...)

Programmbeispiel

```
1 M1 = Min(2,1,3,4,10,100)          'Weist der numerischen Variablen M1 den Wert „1“ zu
```

Erläuterung

- Die Funktion Min berechnet den minimalen Wert der angegebenen numerischen Ausdrücke.
- Die maximale Zeilenlänge wird durch die in einer Zeile zulässige Anzahl von Zeichen begrenzt (240 Zeichen).
- Auf die Argumente <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2> und <...> der Funktion Min darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Max

8.2.30 Mirror$

Funktion: Bits einer Zeichenkette spiegeln

Die Funktion spiegelt die Bits einer Zeichenkette.

Eingabeformat

<Zeichenkettenvariable> = Mirror$ (<Zeichenkette>)

Programmbeispiel

```
1 C1$ = Mirror$("BJ")               'Weist der Zeichenkettenvariablen C1$ den Wert „RB“
                                    zu
                                    „BJ“ = &H42,&H4A = &B0100 0010,&B0100 1010
                                    gespiegelt:  &B0101 0010,&B0100 0010 =
                                                 &H52,&H42
                                    Ausgabe:     „RB“
```

Erläuterung

- Die Funktion spiegelt die binären Bits der Zeichen einer Zeichenkette. Das Ergebnis ist eine Zeichenkette, die dem gespiegelten Wert entspricht.

8.2.31 Mki$

Funktion: 2-Byte-Zeichenkette erzeugen

Die Funktion wandelt einen numerischen Ausdruck vom Typ Integer in eine 2-Byte-Zeichenkette um.

Eingabeformat

<Zeichenkettenvariable> = Mki$ (<Numerischer Ausdruck>)

Programmbeispiel

```
1 C1$ = Mki$(20299)          'Weist der Zeichenkettenvariablen C1$ die
                             Zeichenkette „OK“ zu
2 M1 = Cvi(C1$)              'Weist der numerischen Variablen M1 den Wert
                             „20299“ zu
```

Erläuterung

- Die Funktion wandelt die beiden niederwertigsten Bytes eines numerischen Ausdrucks vom Typ Integer in eine 2-Byte-Zeichenkette um.
- Die Umwandlung einer 2-Byte-Zeichenkette in einen numerischen Ausdruck erfolgt mit der Funktion Cvi.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

Asc, Cvi, Cvs, Cvd, Mks$, Mkd$

8.2.32 Mks$

Funktion: 4-Byte-Zeichenkette erzeugen

Die Funktion wandelt einen numerischen Ausdruck vom Typ Real mit einfacher Genauigkeit in eine 4-Byte-Zeichenkette um.

Eingabeformat

<Zeichenkettenvariable> = Mks$ (<Numerischer Ausdruck>)

Programmbeispiel

```
1  C1$ = Mks$(100.1)          'Weist der Zeichenkettenvariablen C1$ den Wert
                              „100,1“ zu
2  M1 = Cvs(C1$)              'Weist der numerischen Variablen M1 den Wert
                              „100,1“ zu
```

Erläuterung

- Die Funktion wandelt die vier niederwertigsten Bytes eines numerischen Ausdrucks vom Typ Real mit einfacher Genauigkeit in eine 4-Byte-Zeichenkette um.
- Die Umwandlung einer 4-Byte-Zeichenkette in einen numerischen Ausdruck erfolgt mit der Funktion Cvs.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

Asc, Cvi, Cvs, Cvd, Mki$, Mkd$

8.2.33 Mkd$

Funktion: 8-Byte-Zeichenkette erzeugen

Die Funktion wandelt einen numerischen Ausdruck vom Typ Real mit doppelter Genauigkeit in eine 8-Byte-Zeichenkette um.

Eingabeformat

<Zeichenkettenvariable> = Mkd$ (<Numerischer Ausdruck>)

Programmbeispiel

```
1 C1$ = Mkd$(10000.1)        'Weist der Zeichenkettenvariablen C1$ den Wert
                             „10000,1“ zu
2 M1 = Cvd(C1$)              'Weist der numerischen Variablen M1 den Wert
                             „10000,1“ zu
```

Erläuterung

- Die Funktion wandelt die acht niederwertigsten Bytes eines numerischen Ausdrucks vom Typ Real mit doppelter Genauigkeit in eine 8-Byte-Zeichenkette um.
- Die Umwandlung einer 8-Byte-Zeichenkette in einen numerischen Ausdruck erfolgt mit der Funktion Cvd.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

Asc, Cvi, Cvs, Cvd, Mki$, Mks$

8.2.34 PosCq

Funktion: Position prüfen

Die Funktion prüft, ob die angegebene Position innerhalb der zulässigen Verfahrweggrenze liegt.

Eingabeformat

<Numerische Variable> = PosCq (<Positionsvariable>)

Programmbeispiel

```
1 M1 = PosCq(P1)             'Weist der numerischen Variablen eine „1“ zu, falls die
                             Position P1 innerhalb der zulässigen Verfahrweg-
                             grenze liegt
```

Erläuterung

- Liegt die Position innerhalb der zulässigen Verfahrweggrenze, wird die numerische Variable auf „1“ gesetzt, liegt sie außerhalb auf „0“.
- Das Argument kann Positions- oder Gelenkdaten enthalten.

8.2.35 PosMid

Funktion: Mittelposition berechnen

Die Funktion berechnet bei Linear-Interpolation die Mittelposition zwischen zwei Punkten.

Eingabeformat

<Positionsvariable> = PosMid (<Positionsvariable 1>, <Positionsvariable 2>, <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2>)

Programmbeispiel

```
1  P1 = PosMid(P2,P3,0,0)        'Weist der Positionsvariablen P1 die Daten der Position
                                 (inklusive der Stellungsdaten) in der Mitte zwischen
                                 P2 und P3 zu
```

Erläuterung

- Die Positionsvariable 1 gibt den Startpunkt, die Positionsvariable 2 den Endpunkt der Linear-Interpolation an.
- Die numerischen Ausdrücke 1 und 2 entsprechen den TYPE-Angaben des Mvs-Befehls.
- Zwischen dem Start- und Endpunkt muss mit dem angegebenen Interpolationstyp eine Linear-Interpolation ausführbar sein. Weichen z. B. die Stellungsmerker der Start- und der Endposition voneinander ab, erfolgt eine Fehlermeldung.
- Auf die Argumente <Positionsvariable 1>, <Positionsvariable 2>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion PosMid darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.36 PtoJ

Funktion: Positions- in Gelenkdaten umwandeln

Die Funktion wandelt die angegebenen Positionsdaten in Gelenkdaten um.

Eingabeformat

<Gelenkvariable> = PtoJ (<Positionsvariable>)

Programmbeispiel

```
1  J1 = PtoJ(P1)          'Weist die über die Positionsvariable P1 im XYZ-Format
                           festgelegte Position der Gelenkvariablen J1 zu
```

Erläuterung

- Als Argument dürfen keine Gelenkvariablen verwendet werden. Bei Verwendung von Gelenkvariablen erfolgt eine Fehlermeldung.
- Da bei Ausführung der Funktion Gelenkdaten übertragen werden, darf auf der linken Seite der Gleichung keine Positionsvariable verwendet werden. Bei Verwendung einer Positionsvariablen erfolgt eine Fehlermeldung.
- Bei der Software-Version P8 (Serie SD) bzw. N8 (Serie SQ) ist der Mechanismus standardmäßig mit der Nummer „1“ definiert. Wenn nach der Ausführung des RelM-Befehls dem Mechanismus die Nummer „1“ zugeordnet wurde oder wenn die Programmplatznummer (Slot) ungleich „1“ ist, dann ist die Ausführung des GetM-Befehls nicht notwendig. Ist die Mechanismusnummer nicht „1“, muss zuvor der GetM-Befehl ausgeführt werden.

Steht in Beziehung zu folgenden Funktionen:

JtoP

8.2.37 Rad

Funktion: Grad in Radiant umwandeln

Die Funktion wandelt einen in Grad angegebenen Winkel in Radiant um.

Eingabeformat

<Numerische Variable> = Rad (<Numerischer Ausdruck>)

Programmbeispiel

```
1 P1 = P_Curr                'Weist der Positionsvariablen P1 den Wert der aktuellen
                             Position zu
2 P1.C = Rad(90)             'Weist der C-Komponente der Position P1 +90 Grad zu
3 Mov P1                     'Position P1 mittels Gelenk-Interpolation anfahren und
                             den Winkel der C-Achse von der aktuellen Position aus
                             um +90 Grad drehen
```

Erläuterung

- Die Funktion Rad wandelt einen in Grad (deg) angegebenen Winkel in einen Winkel mit der Einheit Radiant (rad) um.
- Die Funktion kann dazu verwendet werden, die Stellungsdaten A, B und C einer Position auf definierte Werte zu setzen. Weiterhin findet sie Anwendung in der Berechnung trigonometrischer Funktionen.

Steht in Beziehung zu folgenden Funktionen:

Deg

8.2.38 Rdfl1

Funktion: Stellungsmerker in Zeichenkette umwandeln

Die Funktion wandelt den Stellungsmerker der festgelegten Position in eine Zeichenkette („R“/„L“, „A“/„B“ und „N“/„F“) um.

Eingabeformat

<Zeichenkettenvariable> = Rdfl1 (<Positionsvariable>, <Numerischer Ausdruck>)

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Positionsvariable>	Legt die Position fest, deren Stellungsmerker in eine Zeichenkette umgewandelt wird
<Numerischer Ausdruck>	Legt fest, welcher Stellungsmerker in eine Zeichenkette umgewandelt wird 0 = „R“/„L“ 1 = „A“/„B“ 2 = „N“/„F“

Programmbeispiel

```
1  P1 = (100,0,100,180,0,180)(7,0)     'Der Stellungsmerker ist auf „7“ (&B111), also RAN gesetzt
2  C1$ = Rdfl1(P1,1)                   'Weist der Zeichenkettenvariablen C1$ das Zeichen „A“ zu
```

Erläuterung

- Argument 1 gibt die Positionsvariable, Argument 2 den Stellungsmerker an, der in eine Zeichenkette umgewandelt wird.
- Die Funktion ist auf den Stellungsmerker FL1 von Positionsdaten anwendbar.
- Auf die Argumente <Positionsvariable> und <Numerischer Ausdruck> der Funktion Rdfl1 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Rdfl2, Setfl1, Setfl2

8.2.39 Rdfl2

Funktion: Multirotationsdaten erzeugen

Die Funktion erzeugt die Multirotationsdaten der festgelegten Gelenkachse.

Eingabeformat

<Numerische Variable> = Rdfl2 (<Positionsvariable>, <Numerischer Ausdruck>)

<Numerische Variable>	Legt eine numerische Variable fest
<Positionsvariable>	Legt die Position fest, deren Multirotationsdaten erzeugt werden
<Numerischer Ausdruck>	Legt das Gelenk fest, dessen Multirotationsdaten erzeugt werden 1 ≤ numerischer Ausdruck ≤ 8

Programmbeispiel

```
1 P1 = (100,0,100,180,0,180)(7,&H00100000)                    '
2 M1 = Rdfl2(P1,6)                                            'Weist der numerischen Variablen M1 den
                                                               Wert „1“ zu
```

Erläuterung

- Die Funktion Rdfl2 erzeugt die Multirotationsdaten des über Argument 2 festgelegten Gelenks. Die Festlegung der Position erfolgt über Argument 1.
- Der Wertebereich der Funktion liegt zwischen –8 und 7.
- Die Funktion ist auf den Stellungsmerker FL2 von Positionsdaten anwendbar.
- Stellungsmerker FL2 ist als 32-Bit-Information aufgebaut. Die Multirotationsdaten jeder der 8 Achsen sind über 4 Bit festgelegt.
- Negative Multirotationswerte zwischen –1 und –8 werden zur Anzeige auf der Teaching Box in den Bereich F bis 8 (4-Bit-hexadezimal mit Vorzeichen) umgewandelt und dargestellt.

		Achse								Angezeigter Wert und Anzahl der Rotationen						
		8	7	6	5	4	3	2	1							
Multirotation von J6 ist +1	FL2	0	0	1	0	0	0	0	0	…	–2	–1	0	+1	+2	…
Multirotation von J6 ist –1	FL2	0	0	F	0	0	0	0	0	…	E	F	0	1	2	…

Tab. 8-1: *Anzeige von Multirotationsdaten auf der Teaching Box*

- Auf die Argumente <Positionsvariable> und <Numerischer Ausdruck> der Funktion Rdfl1 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Befehlen:

JRC

Steht in Beziehung zu folgenden Funktionen:

Rdfl1, Setfl1, Setfl2

8.2.40 Rnd

Funktion: Zufallszahl erzeugen

Die Funktion erzeugt eine Zufallszahl.

Eingabeformat

<Numerische Variable> = Rnd (<Numerischer Ausdruck>)

<Numerische Variable>	Eine numerische Variable im Bereich zwischen 0,0 und 1,0 wird übertragen
<Numerischer Ausdruck>	Legt den Startwert der Zufallszahl fest Bei einer Einstellung auf „0“ werden nachfolgende Zahlen ohne Festlegung des Startwertes generiert.

Programmbeispiel

```
1 Dim MRnd(10)                  'Deklariert die Feldvariable Mrnd als eine Variable mit 10 Elementen
2 C1$ = Right(C_Time,2)         'Setzt den Startwert über den TIMER, um verschiedene Zufallszahlen zu erhalten
3 MRndBS = Cvi(C1)              'Weist der Variablen MRndBS den Integer-Wert der Zeichenkettenvariablen C1 zu
4 MRnd(1) = Rnd(MRndBS)         'Setzt den Startwert und weist dem ersten Element der Feldvariablen die erste Zufallszahl zu
5 For M1 = 2 TO 10              'Erzeugt weitere 9 Zufallszahlen
6    MRnd(M1) = Rnd(0)
7 Next M1
```

Erläuterung

- Die Funktion Rnd erzeugt eine Zufallszahl. Der Startwert wird über das Argument gesetzt.
- Ist das Argument „0“, wird kein Startwert gesetzt und die nächste Zufallszahl erzeugt.
- Bei gleichen Startwerten ergibt sich die gleiche Reihe an Zufallszahlen.

8.2.41 Right$

Funktion: Teil einer Zeichenkette erzeugen

Die Funktion erzeugt einen Teil der angegebenen Zeichenkette, beginnend mit dem rechten Zeichen.

Eingabeformat

<Zeichenkettenvariable> = Right$ (<Zeichenkette>, <Numerischer Ausdruck>)

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Numerischer Ausdruck>	Legt die Anzahl der auszugebenden Zeichen fest

Programmbeispiel

```
1 C1$ = Right$("ABCDEFG",3)        'Weist der Zeichenkettenvariablen C1$ die
                                    Zeichenkette „EFG“ zu
```

Erläuterung

- Die Funktion erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, beginnend mit dem rechten Zeichen, ist im zweiten Argument festgelegt.
- Ist die Anzahl der auszugebenden Zeichen negativ oder größer als die Länge der Zeichenkette, erfolgt eine Fehlermeldung.
- Auf die Argumente <Zeichenkette> und <Numerischer Ausdruck> der Funktion Right$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Left$, Mid$, Len

8.2.42 Setfl1

Funktion: Stellungsmerker ändern

Die Funktion ändert den Stellungsmerker der festgelegten Position über eine Zeichenkette (z. B. „RAN“).

Eingabeformat

<Positionsvariable> = Setfl1 (<Positionsvariable>,<Zeichenkette>)

<Positionsvariable>	Legt die Positionsvariable fest, deren Stellungsmerker geändert werden soll
<Zeichenkette>	Legt fest, welcher Stellungsmerker geändert werden soll Es können mehrere Stellungsmerkerkennungen geändert werden. „R“ oder „L“: rechts/links „A“ oder „B“: oben/unten „N“ oder „F“: nicht kippen/kippen

Programmbeispiel

```
1 Mov P1                    'Positions P1 mittels Gelenk-Interpolation anfahren
2 P2 = Setfl1(P1,"LBF")     'Stellungsmerker der Position P1 auf „links/unten/
                            kippen“ setzen und der Position P2 zuweisen
3 Mov P2                    'Position P2 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Der Stellungsmerker der im Argument 1 angegebenen Position wird auf die im Argument 2 angegebenen Werte der Zeichenkette gesetzt und in die Positionsvariable links vom Gleichheitszeichen übertragen.
- Die Funktion ändert nur die Daten des Stellungsmerkers FL1. Die durch das Argument 1 festgelegten Positionsdaten (X, Y, Z, A, B, C und FL2) bleiben unverändert.
- Der Stellungsmerker wird über die Zeichenkette, beginnend mit dem letzten Zeichen, festgelegt. Ist z. B. die Zeichenkette „LR“ vorgegeben, ergibt sich für den resultierenden Stellungsmerker „L“.
- Die Änderung eines Stellungsmerkers über einen numerischen Wert kann nach folgendem Schema erfolgen: P1.FL1 = 7.
- Die Stellungsmerker haben in Abhängigkeit vom Robotermodell unterschiedliche Bedeutungen. Eine detaillierte Beschreibung finden Sie im Technischen Handbuch des jeweiligen Roboters.

In der Positionskonstante (100, 0, 300, 180, 0, 180) (7, 0) ist der Stellungsmerker auf „7“ gesetzt. Die aktuelle Position wird über ein Bitmuster dargestellt.

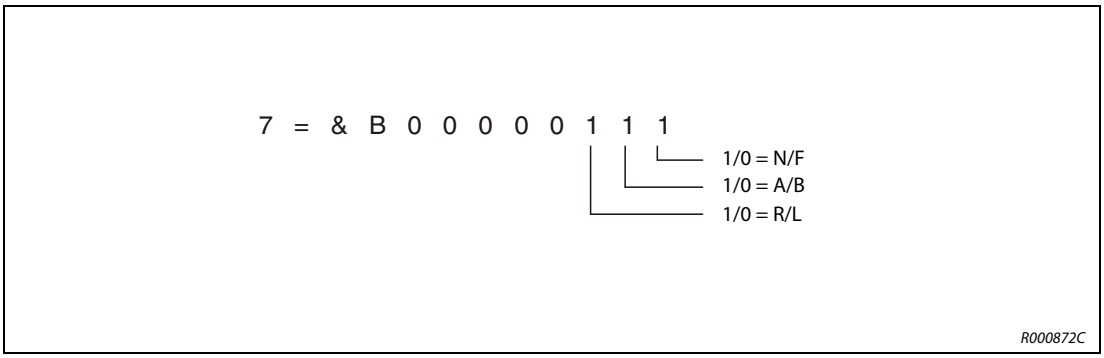


Abb. 8-2: *Bedeutung der Stellungsmerker*

- Auf die Argumente <Positionsvariable> und <Zeichenkette> der Funktion Setfl1 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Rdfl1, Rdfl2, Setfl2

8.2.43 Setfl2

Funktion: Multirotationsdaten ändern

Die Funktion ändert die Multirotationsdaten der festgelegten Position.

Eingabeformat

<Positionsvariable> = Setfl2 (<Positionsvariable>, <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2>)

<Positionsvariable>	Legt die Positionsvariable fest, deren Multirotationsdaten geändert werden sollen
<Numerischer Ausdruck 1>	Legt das Gelenk fest, dessen Multirotationsdaten geändert werden sollen 1 ≤ numerischer Ausdruck 1 ≤ 8
<Numerischer Ausdruck 2>	Legt die Multirotationsdaten fest –8 ≤ numerischer Ausdruck 2 ≤ 7

Programmbeispiel

```
1 Mov P1                    'Position P1 mittels Gelenk-Interpolation anfahren
2 P2 = Setfl2(P1,6,1)       'Multirotationsdaten des Gelenks 6 der Position P1 auf
                            „1“ setzen
3 Mov P2                    'Position P2 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Multirotationsdaten des im Argument 2 angegebenen Gelenks der Positionsvariablen werden auf die im Argument 3 angegebenen Werte gesetzt und in die Positionsvariable links vom Gleichheitszeichen übertragen.
- Die Funktion ändert nur die Daten des Stellungsmerkers FL2. Die durch das Argument 1 festgelegten Positionsdaten (X, Y, Z, A, B, C und FL1) bleiben unverändert.

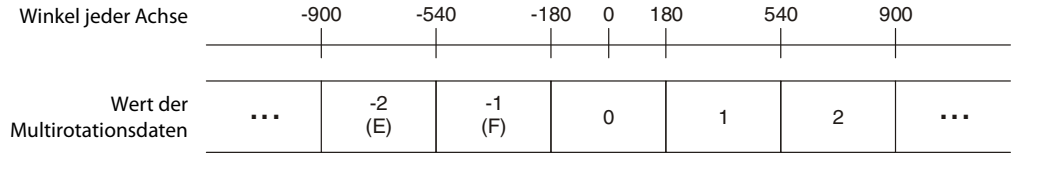


Abb. 8-3: *Wert der Multirotationsdaten*

- Auf die Argumente <Positionsvariable>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion Setfl2 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

Rdfl1, Rdfl2, Setfl1

8.2.44 SetJnt

Funktion: Gelenkvariable ändern

Die Funktion ändert die Werte der festgelegten Gelenkvariablen.

Eingabeformat

```
<Gelenkvariable> = SETJNT (<J1-Achse>[,<J2-Achse>[,<J3-Achse>[,<J4-Achse>
                          [,<J5-Achse>[,<J6-Achse>[,<J7-Achse>[,<J8-Achse>]]]]]]])
```

<Gelenkvariable>	Legt eine Gelenkvariable fest
<J1-Achse> bis <J8-Achse>	Die Einheit der Achsendaten ist RAD (Für direkt angetriebene Achsen ist die Einheit mm.)

Programmbeispiel

```
1  J1 = J_Curr                          'Überträgt die Gelenkdaten der aktuellen Position in
                                        die Gelenkvariable J1
2  For M1 = 0 to 60 STEP 10
3     M2 = J1.J3 + Rad(M1)
4     J2 = SetJnt(J1.J1,J1.J2,M2)       'Erhöht den Wert der J3-Achse bei jedem Schleifen-
                                        durchlauf um 10 Grad. Die Werte der J4-Achse und der
                                        nachfolgenden Achsen bleiben unverändert.
5     Mov J2                            'Position J2 mittels Gelenk-Interpolation anfahren und
                                        den Winkel der J3-Achse von der aktuellen Position aus
                                        um 10 Grad drehen
6  Next M1                              'Sprung zu Zeile 2
7  M0 = Rad(0)                          Weist der numerischen Variablen M0 den Wert
                                        0 Grad zu
8  M90 = Rad(90)                        'Weist der numerischen Variablen M90 den Wert
                                        90 Grad zu
9  J3 = SetJnt(M0,M0,M90,M0,M90,M0)     'Festlegung der Gelenkvariablen J3
10 Mov J3                               'Position J3 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Funktion ermöglicht die Änderung der Winkel jeder einzelnen Achse einer Gelenkvariablen.
- Die Gelenkvariable kann durch Argumente beschrieben werden.
- Außer dem Argument für die J1-Achse können alle Argumente weggelassen werden. Dabei müssen auch alle nachfolgenden Argumente weggelassen werden. Beschreibungen wie z. B. SetJnt(10,10,,,,10) sind nicht erlaubt.
- Auf die Argumente der Funktion SetJnt darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

SetPos

Steht in Beziehung zu folgenden Parametern:

AXUNT, PRGMDEG

8.2.45 SetPos

Funktion: Positionsvariable ändern

Die Funktion ändert die Werte der festgelegten Positionsvariablen.

Eingabeformat

```
<Positionsvariable> = SetPos (<X-Achse>[,<Y-Achse>[,<Z-Achse>[,<A-Achse>[,<B-Achse>
                              [,<C-Achse>[,<L1-Achse>[,<L2-Achse>]]]]]])
```

<Positionsvariable>	Legt eine Positionsvariable fest
<X-Achse> bis <Z-Achse>	Die Einheit der Achsendaten ist mm
<A-Achse> bis <C-Achse>	Die Einheit der Achsendaten ist Rad (Die Einheit kann über Parameter PRGMDeg auf „DEG“ geändert werden.)
<L1-Achse> und <L2-Achse>	Die Einheit hängt von der Einstellung des Parameters AXUNT ab.

Programmbeispiel

```
1  P1 = P_Curr                    'Überträgt die aktuelle Position in die
                                  Positionsvariable P1
2  For M1 = 0 to100 STEP 10
3     M2 = P1.Z + M1
4     P2 = SetPos(P1.X,P1.Y,M2)   'Erhöht den Wert der Z-Achse bei jedem Schleifen-
                                  durchlauf um 10 mm. Die Werte der A-Stellungsdaten
                                  und der nachfolgenden Daten bleiben unverändert.
5     Mov P2                      'Position P2 mittels Gelenk-Interpolation anfahren und
                                  den Wert der Z-Achse von der aktuellen Position aus
                                  um 10 mm verschieben
6  Next M1                        'Sprung zu Zeile 2
```

Erläuterung

- Die Funktion ermöglicht die Änderung der Werte jeder einzelnen Komponente einer Positionsvariablen.
- Die Positionsvariable kann durch Argumente beschrieben werden.
- Außer dem Argument für die X-Achse können alle Argumente weggelassen werden. Dabei müssen auch alle nachfolgenden Argumente weggelassen werden. Beschreibungen wie z. B. SetPos(10,10,,,,10) sind nicht erlaubt.
- Auf die Argumente der Funktion SetPos darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

SetJnt

Steht in Beziehung zu folgenden Parametern:

AXUNT, PRGMDEG

8.2.46 Sgn

Funktion: Vorzeichen prüfen

Die Funktion prüft das Vorzeichen des angegebenen numerischen Ausdrucks.

Eingabeformat

<Numerische Variable> = Sgn (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = –1                    'Weist der numerischen Variablen M1 den Wert „–1“ zu
2 M2 = Sgn(M1)               'Weist der numerischen Variablen M2 den Wert „–1“ zu
```

Erläuterung

- Die Funktion Sgn prüft das Vorzeichen eines numerischen Ausdrucks nach folgendem Schema:

  Positiver Wert → 1
  Null → 0
  Negativer Wert → –1

8.2.47 Sin

Funktion: Sinus berechnen

Die Funktion berechnet den Sinus.

Eingabeformat

<Numerische Variable> = Sin (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Sin(Rad(60))          'Weist der numerischen Variablen M1 den Wert
                             „0,866025“ zu
```

Erläuterung

- Die Funktionen berechnet den Sinus des numerischen Ausdrucks. Die Einheit des Arguments ist Radiant.
- Als Definitionsbereich ist der gesamte gültige Zahlenbereich zugelassen.
- Der Wertebereich der Funktion Sin ist –1 bis 1.

Steht in Beziehung zu folgenden Funktionen:

Cos, Tan, Atn/Atn2

8.2.48 Sqr

Funktion: Quadratwurzel berechnen

Die Funktion berechnet die Quadratwurzel.

Eingabeformat

<Numerische Variable> = Sqr (<Numerischer Ausdruck>)

Programmbeispiel

```
1  M1 = Sqr(2)                    'Weist der numerischen Variablen M1 den Wert
                                  „1,414214“ zu
```

Erläuterung

- Die Funktion berechnet die Quadratwurzel des angegebenen numerischen Ausdrucks.
- Bei negativem Argument erfolgt eine Fehlermeldung.

8.2.49 Strpos

Funktion: Zeichenkette suchen

Die Funktion sucht die angegebene Zeichenkette innerhalb einer anderen Zeichenkette.

Eingabeformat

<Numerische Variable> = Strpos (<Zeichenkette 1>, <Zeichenkette 2>)

Programmbeispiel

```
1  M1 = Strpos("ABCDEFG","DEF")   'Weist der numerischen Variablen M1 den Wert „4“ zu
```

Erläuterung

- Die Funktion Strpos ermittelt den Wert des ersten Auftretens der Zeichenkette 2 innerhalb der Zeichenkette 1.
- Ist die Länge der Zeichenkette 2 gleich 0, erfolgt eine Fehlermeldung.
- Kann die gesuchte Zeichenkette nicht gefunden werden, ist das Suchergebnis „0“.
- Auf die Argumente <Zeichenkette 1> und <Zeichenkette 2> der Funktion Strpos darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.50 Str$

Funktion: Zahl in Zeichenkette umwandeln

Die Funktion wandelt eine Zahl in eine Zeichenkette um.

Eingabeformat

<Zeichenkettenvariable> = Str$ (<Numerischer Ausdruck>)

Programmbeispiel

```
1 C1$ = Str$(123)          'Weist der Zeichenkettenvariablen C1$ die
                           Zeichenkette „123“ zu
```

Erläuterung

- Die Funktion wandelt den angegebenen numerischen Ausdruck in eine Zeichenkette um.
- Die Umkehrung der Funktion Str$ erfolgt über die Funktion Val.

Steht in Beziehung zu folgenden Funktionen:

Bin$, Hex$, Val

8.2.51 Tan

Funktion: Tangens berechnen

Die Funktion berechnet den Tangens.

Eingabeformat

<Numerische Variable> = Tan (<Numerischer Ausdruck>)

Programmbeispiel

```
1 M1 = Tan(Rad(60))        'Weist der numerischen Variablen M1 den Wert
                           „1,732051“ zu
```

Erläuterung

- Die Funktion berechnet den Tangens des numerischen Ausdrucks. Die Einheit des Arguments ist Radiant.
- Als Definitionsbereich ist der gesamte gültige Zahlenbereich zugelassen.
- Der Wertebereich der Funktion ist der gesamte gültige Zahlenbereich.

Steht in Beziehung zu folgenden Funktionen:

Sin, Cos, Atn/Atn2

8.2.52 Val

Funktion: Zeichenkette in Zahl umwandeln

Die Funktion wandelt eine Zeichenkette in eine Zahl um.

Eingabeformat

<Numerische Variable> = Val (<Zeichenkette>)

Programmbeispiel

```
1  M1 = Val("15")          'Weist der numerischen Variablen M1 den Wert der
                           Zeichenkette „15“ zu
2  M2 = Val("&B1111")      'Weist der numerischen Variablen M2 den Wert der
                           Zeichenkette „&B1111“ zu
3  M3 = Val("&HF")         'Weist der numerischen Variablen M3 den Wert der
                           Zeichenkette „&HF“ zu
```

Erläuterung

- Die Funktion wandelt die angegebene Zeichenkette in einen numerischen Ausdruck um.
- Die Zeichenkette kann binär (&B) und hexadezimal (&H) dargestellt werden.
- Im Programmbeispiel wird in alle Variablen M1, M2 und M3 der Wert „15“ übertragen.

Steht in Beziehung zu folgenden Funktionen:

Bin$, Hex$, Str$

8.2.53 Zone

Funktion: Position prüfen

Die Funktion prüft, ob die Position innerhalb eines durch zwei Punkte definierten Quaders liegt.

Eingabeformat

<Numerische Variable> = Zone (<Position 1>, <Position 2>, <Position 3>)

<Position 1>	Legt die Position fest, deren Lage geprüft werden soll
<Position 2>	Legt die erste Position zur Bereichsdefinition fest
<Position 3>	Legt die zweite Position zur Bereichsdefinition fest

Programmbeispiel

```
1  M1 = Zone(P1,P2,P3)          'Weist der numerischen Variablen M1 das Prüfergebnis zu
2  If M1 = 1 Then Mov P_Safe Else End   'Fährt die Position P_Safe an, falls die Position P1 innerhalb des durch die Positionen P2 und P3 definierten Quaders liegt und beendet das Programm, falls die Position außerhalb liegt
```

Erläuterung

- Die Funktion Zone prüft, ob die Position 1 innerhalb eines durch die Positionen 2 und 3 definierten Quaders liegt. Die Positionen 2 und 3 bilden zwei Punkte auf der Diagonalen des erzeugten Quaders. Liegt die Position 1 innerhalb des Quaders, wird eine „1“, ansonsten eine „0“ in die numerische Variable übertragen.

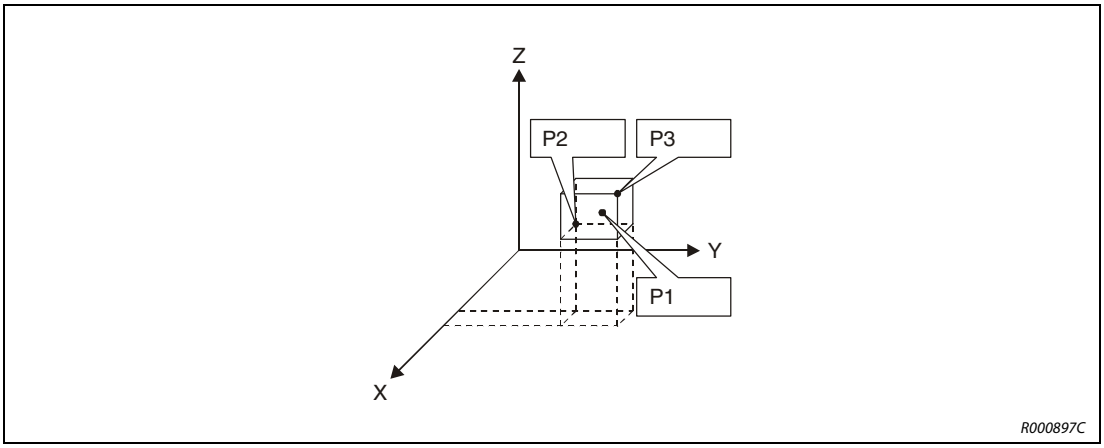


Abb. 8-4: *Definition eines quaderförmigen Prüfbereiches*

- Das Prüfergebnis wird ermittelt, indem für jedes Element der Position 1 (X, Y, Z, A, B, C, L1 und L2) untersucht wird, ob es zwischen den entsprechenden Werten der Positionen 2 und 3 liegt.
- Bei den Stellungsdaten (A, B und C) wird geprüft, ob die Werte der Position 1 innerhalb des Bereichs liegen, der bei Rotation vom Winkel der Position 2 zum Winkel der Position 3 in positiver Richtung durchfahren wird.

Beispiel ▽ Sind die A-Komponenten der Position 2 und 3: P2.A = –100, P3.A = +100, liegt die Position 1 mit der A-Komponente P1.A = 50 innerhalb des Bereiches. Diese Prüfung wird in gleicher Weise auch für die B- und C- Komponente durchgeführt.

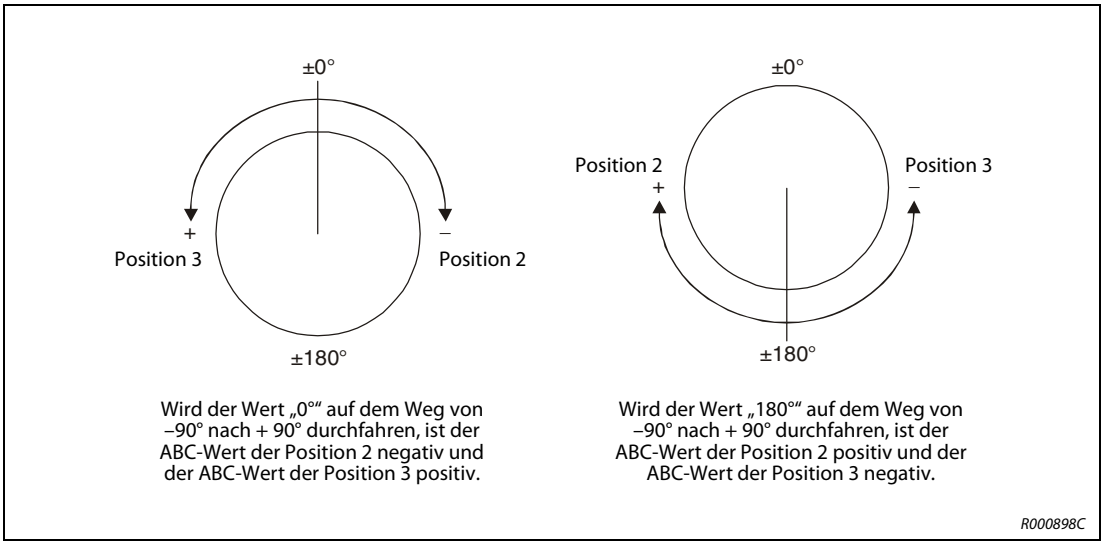

±0°
+
Position 3
–
Position 2
±180°
Wird der Wert „0°" auf dem Weg von –90° nach + 90° durchfahren, ist der ABC-Wert der Position 2 negativ und der ABC-Wert der Position 3 positiv.
±0°
Position 2
+
Position 3
–
±180°
Wird der Wert „180°" auf dem Weg von –90° nach + 90° durchfahren, ist der ABC-Wert der Position 2 positiv und der ABC-Wert der Position 3 negativ.
R000898C

Abb. 8-5: *Prüfung der Stellungsdaten*

△

- Komponenten, die nicht geprüft werden sollen oder fehlen, müssen auf folgende Werte gesetzt werden:

  Bei Einheit Grad:

  – Position 2 wird auf –360° gesetzt
  – Position 3 wird auf 360° gesetzt

  Bei Einheit mm:

  – Position 2 wird auf –10000 gesetzt
  – Position 3 wird auf 10000 gesetzt

- Auf die Argumente <Position 1>, <Position 2> und <Position 3> der Funktion Zone darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.54 Zone2

Funktion: Position prüfen

Die Funktion prüft, ob die Position innerhalb eines durch zwei Punkte definierten Zylinders liegt.

Eingabeformat

<Numerische Variable> = Zone2 (<Position 1>, <Position 2>, <Position 3>, <Numerischer Ausdruck>)

<Position 1>	Legt die Position fest, deren Lage geprüft werden soll
<Position 2>	Legt die erste Position zur Bereichsdefinition fest
<Position 3>	Legt die zweite Position zur Bereichsdefinition fest
<Numerischer Ausdruck>	Legt den Radius des Bereichs an beiden Endpunkten fest

Programmbeispiel

```
1 M1 = Zone2(P1,P2,P3,50)          'Weist der numerischen Variablen M1 das Prüfergebnis zu
2 If M1 = 1 Then Mov P_Safe Else End   'Fährt die Position P_Safe an, falls die Position P1 innerhalb des durch die Positionen P2 und P3 definierten Zylinders liegt und beendet das Programm, falls die Position außerhalb liegt
```

Erläuterung

- Die Funktion Zone2 prüft, ob die Position 1 innerhalb eines durch die Positionen 2 und 3 definierten Zylinders liegt. Die Positionen 2 und 3 bilden zwei Endpunkte des Zylinders. Der numerische Ausdruck legt einen Radius um diese Eckpunkte fest. Liegt die Position 1 innerhalb des Rechtecks, wird eine „1“, ansonsten eine „0“ in die numerische Variable übertragen.
- Die Funktion prüft, ob die XYZ-Komponenten einer Position innerhalb des festgelegten Bereiches liegen. Die Stellungsdaten werden nicht geprüft.

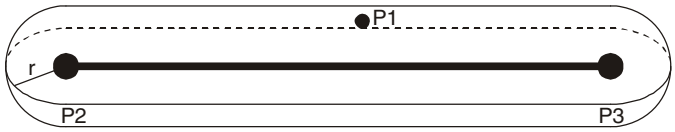


Abb. 8-6: *Definition des zylindrischen Prüfbereiches*

- Auf die Argumente <Position 1>, <Position 2>, <Position 3> und <Numerischer Ausdruck> der Funktion Zone2 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.55 Zone3

Funktion: Position prüfen

Die Funktion prüft, ob die Position innerhalb eines durch drei Punkte definierten Quaders liegt.

Eingabeformat

<Numerische Variable> = Zone3 (<Position 1>, <Position 2>, <Position 3>, <Position 4>, <Numerischer Ausdruck W>, Numerischer Ausdruck H>, <Numerischer Ausdruck L>)

<Position 1>	Legt die Position fest, deren Lage geprüft werden soll
<Position 2>	Legt die erste Position zur Bereichsdefinition fest
<Position 3>	Legt die zweite Position zur Bereichsdefinition fest
<Position 4>	Legt die dritte Position zur Definition der Ebene fest, die den Bereich mit <Position 2> und <Position 3> darstellt
<Numerischer Ausdruck W>	Legt die Breite des Quaders fest, der den Bereichs darstellt
<Numerischer Ausdruck H>	Legt die Höhe des Quaders fest, der den Bereichs darstellt
<Numerischer Ausdruck L>	Legt jeden Tiefenabstand von <Position 2> und <Position 3> zum Quader fest, der den Bereich darstellt

Programmbeispiel

```
1 M1=Zone3(P1,P2,P3,P4,100,100,50)   'Weist der numerischen Variablen M1 das Prüfergebnis zu
2 If M1 = 1 Then Mov P_Safe Else End 'Fährt die Position P_Safe an, falls die Position P1 innerhalb des durch die Positionen P2 und P3 definierten Zylinders liegt und beendet das Programm, falls die Position außerhalb liegt
```

Erläuterung

- Die Funktion Zone3 prüft, ob sich die Position 1 innerhalb eines durch die Positionen 2, 3 und 4, sowie durch die numerischen Ausdrücke W, H und L definierten Quaders befindet. Liegt die Position 1 innerhalb dieses Raums, wird eine „1“, ansonsten eine „0“ in die numerische Variable übertragen.
- Die Funktion prüft, ob die XYZ-Komponenten einer Position innerhalb des festgelegten Bereiches liegen. Die Stellungsdaten werden nicht geprüft.

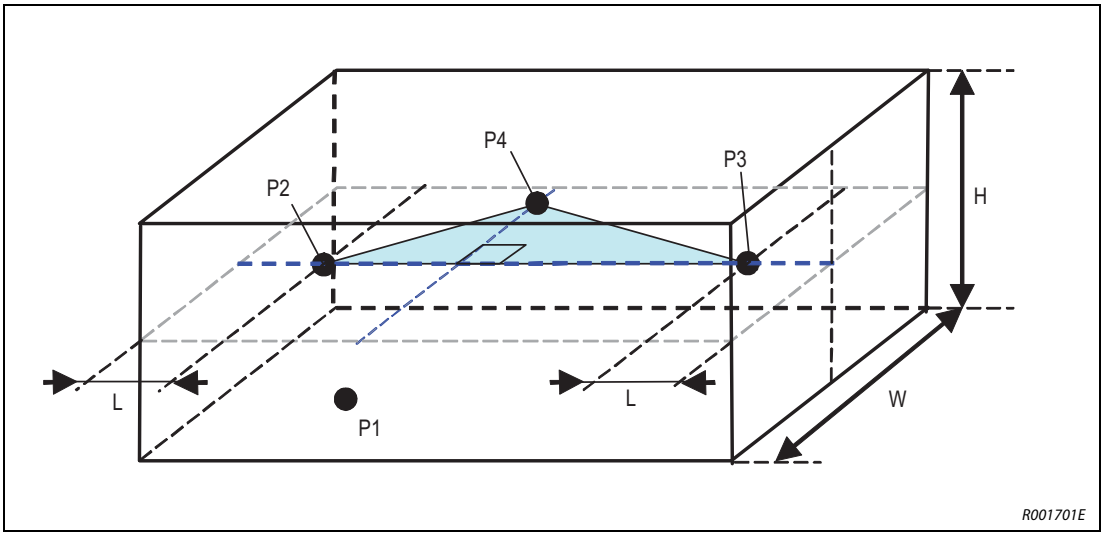


Abb. 8-7: *Definition des quaderförmigen Prüfbereiches*

- Auf die Argumente <Position 1>, <Position 3>, <Position 4>, <Numerischer Ausdruck W>, <Numerischer Ausdruck H> und <Numerischer Ausdruck L> der Funktion Zone3 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.
- Bei negativem Argument für <Numerischer Ausdruck W> und <Numerischer Ausdruck H> erfolgt eine Fehlermeldung.
- Der festgelegte Bereich kann nicht erzeugt werden, wenn für <Position 2> bis <Position 4> die gleiche Positionen bzw. Positionen eingegeben werden, die auf einer Geraden liegen. In diesem Fall wird ohne Prüfung eine „–1“ übertragen.
- Wird in <Numerischer Ausdruck L> eine negative Zahl eingegeben und beträgt deren Absolutwert weniger als die Hälfte des Abstands zwischen <Position 2> und <Position 3>, wird ohne Prüfung eine „–1“ übertragen.

9 Parameter

9.1 Allgemeines

Folgende Tabelle zeigt die anwendungsspezifische Einteilung der Parameter der Steuergeräte CRnD und CRnQ. Mit Hilfe der Parameter können verschiedene Funktionen und Grundeinstellungen beeinflusst werden.

Anwendung	Beschreibung	Seite
Bewegungsparameter	Diese Parameter dienen der Einstellung des Bewegungsbereiches, des Koordinatensystems und auf die Hand bezogener Größen.	9-2
Signalparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung von Signalen.	9-17
Betriebsparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung des Steuergeräte- und des Teaching-Box-Betriebs usw.	9-24
Befehlsparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Programmausführung.	9-27
Kommunikationsparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Kommunikation.	9-31

Tab. 9-1: *Anwendungsspezifische Einteilung der Parameter*

Parameter zur Steuerung von Ein- und Ausgängen sind in Kap. 10 erläutert.

Die Änderung einer Parametereinstellung wird erst nach Aus- und Wiedereinschalten der Spannungsversorgung des Steuergerätes aktiv.

Eine genaue Beschreibung der Vorgehensweise zur Einstellung von Parametern finden Sie in Abschn. 3.18 „Parametermenü“.

GEFAHR:

Nehmen Sie Parametereinstellungen nur mit besonderer Sorgfalt vor. Prüfen Sie die Auswirkungen einer neuen Einstellung auf die beeinflusste Funktion. Fehlerhaft eingestellte Parameter können zu undefinierten Aktivitäten des Roboters führen. Es besteht Verletzungsgefahr!

9.2 Bewegungsparameter

Diese Parameter dienen der Einstellung des Bewegungsbereiches, des Koordinatensystems und der auf die Hand bezogenen Größen.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Verfahrweg-grenzen für Gelenkbewe-gungen	MEJAR	Reelle Zahl 16	Legt die Verfahrweggrenzen für jedes einzelne Gelenk fest Um eine Begrenzung durch die mechanischen Anschläge zu vermeiden, wird eine Überschreitung der Verfahrweggrenzen nicht empfohlen. (–J1, +J1, –J2, +J2, … –J8, +J8) Einheit: Grad	Abhängig vom Mechanismus
Verfahrweg-grenzen für XYZ-Bewegun-gen	MEPAR	Reelle Zahl 6	Legt die Verfahrweggrenzen für das XYZ-Koordinatensystem fest Der Parameter kann dazu verwendet werden, Kollisionen des Roboters mit umliegenden Einrichtungen zu verhindern. (–X, +X, –Y, +Y, –Z, +Z) Einheit: mm	(–X, +X, –Y, +Y, –Z, +Z) = –10000, 10000, –10000, 10000, –10000, 10000
Standard-Werkzeug-koordinaten (siehe auch Abschn. 5.2 und Abschn. 9.7)	MEXTL	Reelle Zahl 6	Legt den Werkzeugmittelpunkt TCP fest Verwenden Sie diesen Parameter, wenn Sie einen Handgreifer installieren und der Werkzeugmittelpunkt angepasst werden muss. Der Parameter ermöglicht eine Überwachung der Stellung an der Handspitze für den XYZ- oder den Werkzeug- JOG-Betrieb. (X, Y, Z, A, B, C) Einheit: mm oder Grad	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Werkzeug-koordinaten 1 bis 16 (siehe auch Abschn. 7.2.48)	MEXTL1 : MEXTL16	Reelle Zahl 6	Diese Werkzeugdaten sind wirksam, wenn die Variable M_Tool auf „1“ bis „16“ gesetzt ist.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Standard-Basis-koordinaten (siehe auch Abschn. 5.2 und Abschn. 9.8)	MEXBS	Reelle Zahl 6	Legt das Basiskoordinatensystem in Beziehung zum Weltkoordinatensystem fest In der Werkseinstellung sind Roboter- und Weltkoordinatensystem identisch. Eine Einstellung des Parameters ist nur dann nötig, wenn der Aufbau der gesamten Anlage verändert wird oder, wenn die gesamte Anlage auf ein Koordinatensystem bezogen ist. (X, Y, Z, A, B, C) Einheit: mm oder Grad	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0

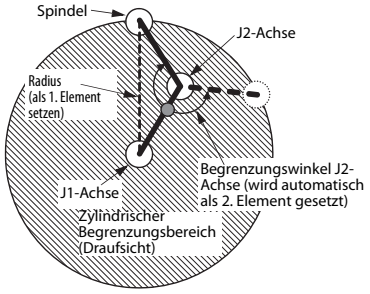
Tab. 9-2: *Übersicht der Bewegungsparameter (1)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Nummer des aktuellen Basis-koordinaten-systems (siehe auch Abschn. 5.2)	MEXBSNO	Reelle Zahl 1	Legt das aktuelle Basiskoordinatensystem über eine Nummer fest (Basiskonvertierung) Auch die aktuellen Werte können angezeigt werden. Beschreibung der Werte: 0: Festlegung des Standardwert des Systems P_NBase (Da P_NBase = (0, 0, 0, 0, 0, 0), werden die Basis-Konvertierungsdaten gelöscht.) 1–8: Festlegung eines Werkstück-koordinatensystems 1 bis 8 (Parameter WK1CORD bis WK8CORD) –1: Die Einstellung der Basis-Transformati-onsdaten wurde auf eine andere Weise vorgenommen (z. B. über den Base-Befehl oder den Parameter MEXBS). HINWEIS: Bei der Einstellung –1kann der Wert nur gele-sen werden.	–1
Benutzerdefinierte Verfahrweggrenze			Es werden benutzerdefinierte Bereiche (maximal 32) festgelegt. Wenn der Roboter in diese Bereiche eindringt, wird das als Ver-fahrwegüberschreitung definiert und als Reaktion darauf kann ein korrespondieren-des Signal geschaltet werden.	
	AREA1CS : AREA32CS	Ganze Zahl 1	Festlegung des Koordinatensystems des benutzerdefinierten Bereichs ① 0: Basiskoordinatensystem 1: Weltkoordinatensystem HINWEIS: Diese Funktion kann ab den folgenden Soft-ware-Versionen genutzt werden. Mit Aus-nahme der folgenden Versionen ist standardmäßig das Basiskoordinatensystem eingestellt. Serie SD: ab Version P1a Serie SQ: ab Version N7A	(X, Y, Z, A, B, C) = 0.0,0.0,0.0, –360.0, –360.0,–360.0
	AREA1P1 : AREA32P1	Reelle Zahl 8	Festlegung der Positionskoordinaten des 1. Palettenеckpunkts des benutzerdefinierten Bereichs n und der Koordinaten der Stel-lungsdaten/Zusatzachsen Setzen Sie die Elemente in folgender Reihen-folge: X, Y, Z, A, B, C, L1 und L2 **Element** / **Beschreibung** / **Einheit** X, Y, Z / Positionskoordinaten des Palettenеckpunkts 1 / mm A, B, C / Stellungsbereich / Grad L1, L2 / Bereich der Zusatzachsen / mm, Grad ① HINWEISE: Legen Sie die Werte in dem Koordinaten-system fest, welches mit AREAnCS definiert wurde. Geben Sie für die Koordinaten A, B und C –360 ein, wenn die Stellungsdaten nicht auf Verfahrwegüberschreitung geprüft werden sollen. Definieren Sie die Elemente L1 und L2 bei der Verwendung von Zusatzachsen. Die mit den Elementen X, Y, Y, Z, L1 und L2 definierten Bereiche bleiben unverändert, auch wenn Änderungen in AREAnP2 erfol-gen.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0, –360.0, –360.0,–360.0,0,0

Tab. 9-2: *Übersicht der Bewegungsparameter (2)*

<table>
<tr><th colspan="2">Parameter</th><th>Anzahl der Felder/ Zeichen</th><th>Beschreibung</th><th>Werkseinstellung</th></tr>
<tr><td rowspan="3">Benutzer-definierte Verfahrweg-grenze</td><td>AREA1P2
:
AREA32P2</td><td>Reelle Zahl
8</td><td>Festlegung der Positionskoordinaten des 2. Palletteneckpunkts des benutzerdefinierten Bereichs n und der Koordinaten der Stellungsdaten/Zusatzachsen
Setzen Sie die Elemente in folgender Reihenfolge: X, Y, Z, A, B, C, L1 und L2
<table><tr><th>Element</th><th>Beschreibung</th><th>Einheit</th></tr><tr><td>X, Y, Z</td><td>Positionskoordinaten des Palletteneckpunkts 2</td><td>mm</td></tr><tr><td>A, B, C</td><td>Stellungsbereich</td><td>Grad</td></tr><tr><td>L1, L2</td><td>Bereich der Zusatzachsen</td><td>mm, Grad</td></tr></table>HINWEISE:
Legen Sie die Werte in dem Koordinaten-system fest, welches mit AREAnCS definiert wurde.
Geben Sie für die Koordinaten A, B und C +360 ein, wenn die Stellungsdaten nicht auf Verfahrwegüberschreitung geprüft werden sollen.
Definieren Sie die Elemente L1 und L2 bei der Verwendung von Zusatzachsen.
Die mit den Elementen X, Y, Y, Z, L1 und L2 definierten Bereiche bleiben unverändert, auch wenn Änderungen in AREAnP1 erfol-gen.</td><td>(X, Y, Z, A, B, C) = 0.0,0.0,0.0, –360.0, –360.0,–360.0,0,0</td></tr>
<tr><td>AREA1ME
:
AREA32ME</td><td>Ganze Zahl
1</td><td>Zuweisung der Begrenzungsbereiche an die Mechanismen 1 bis 3
Standardmäßig ist der Wert auf „1“ gesetzt.
0: Deaktiviert (Keine Bereichsprüfung)
1: Mechanismus 1 (Standardeinstellung)
2: Mechanismus 2
3: Mechanismus 3</td><td>0</td></tr>
<tr><td>AREA1AT
:
AREA32AT</td><td>Ganze Zahl
1
Außerhalb der Verfahrweg-grenze</td><td>Festlegung der Bereichsprüfmethode:
0: Gesperrt
1: Ausgabe eines Im-Bereich-Signals
2: Fehler
<table><tr><th rowspan="2">Einstel-lung</th><th colspan="2">Begrenzungsbereich</th></tr><tr><th>Innerhalb</th><th>Außerhalb</th></tr><tr><td rowspan="2">Signal-ausgabe</td><td>Entsprechendes Im-Bereich-Signal schaltet ein ①</td><td>Entsprechendes Im-Bereich-Signals schaltet aus</td></tr><tr><td>Einschalten des zugehörigen Bits der Statusvari-ablen ②</td><td>Absschalten des zugehörigen Bits der Statusvari-ablen</td></tr><tr><td>Fehler-ausgabe</td><td>Stopp bei Fehler-ausgabe (Fehler H2090)</td><td>—</td></tr></table>① Signalnummer wird über Parameter USRAREA definiert
② Roboterstatusvariable (M_Uar, M_Uar32)
HINWEIS:
Bei Auswahl der Fehlerausgabe, wird nur der Bereich der Positionskoordinaten überprüft, die Stellungsdaten und Zusatzachsen wer-den ignoriert.</td><td>0: Gesperrt</td></tr>
</table>

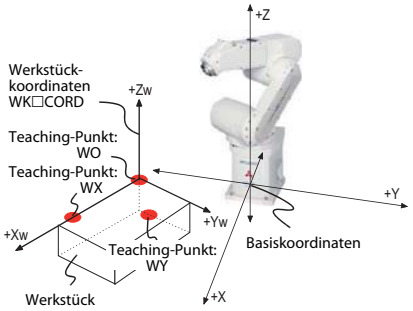
Tab. 9-2: *Übersicht der Bewegungsparameter (3)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Verfahrwegbegrenzungsebene (siehe auch Abschn. 9.10)			Die Verfahrweggrenzen werden über eine Ebene definiert. Die Ebene wird über die Koordinaten X1, Y1, Z1 bis X3, Y3, Z3 festgelegt. Bei Überschreitung dieser Bereichsgrenzen erfolgt eine Fehlermeldung. Folgende 3 Parametertypen können verwendet werden:	
	SFC1P : SFC8P	Reelle Zahl 9	Über SFC1P bis SFC8P können 8 Begrenzungsebenen definiert werden. Setzen Sie die dazu nötigen 9 Elemente in folgender Reihenfolge: X1, Y1, Z1: Flächenursprung X2, Y2, Z2: Position auf der X-Achse in der Ebene X3, Y3, Z3: Position in positiver Richtung auf der Y-Achse in der Ebene	(X1, Y1, Z1, X2, Y2, Z2, X3, Y3, Z3) = 0.0,0.0,0.0,0.0,0.0,0.0, 0.0,0.0,0.0
	SFC1ME : SFC8ME	Ganze Zahl 1	Zuweisung der Begrenzungsebenen an die Mechanismen 1 bis 3 Standardmäßig ist der Wert auf „0“ gesetzt.	0
	SFC1AT : SFC8AT	Ganze Zahl 1	Freigabe der Begrenzungsebenen: 0: gesperrt 1: freigegeben (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der auch der Nullpunkt des Basiskoordinatensystems liegt.) –1: freigegeben (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der der Nullpunkt des Basiskoordinatensystems nicht liegt.)	0 (gesperrt)
Zylindrischer Begrenzungsbereich Funktion steht in folgenden Software-Versionen zur Verfügung: SD-Serie: ab Version S1j SQ-Serie: ab Version R1j Nur für die Roboter: RH-3SDHR/RH-3SQHR			Der Bewegungsbereich des Roboters kann durch einen zylindrischen Bereich mit der J1-Achse als Mittelpunkt eingeschränkt werden. Stellen Sie als Radius den Abstand zur J1-Achse ein.	
	MECAR	Reelle Zahl 2	Aufbau des Parameters: (Radius, Begrenzungswinkel J2-Achse) Stellen Sie den Radius als 1. Element ein. Einheit: mm (zweistellige Dezimalzahl) Ist der Radius „0“, ist die Funktion deaktiviert. Bei Eingabe eines negativen oder nicht erreichbaren Wertes erfolgt eine Fehlermeldung. Der Begrenzungszylinder schränkt den Bewegungsbereich der J2-Achse ein. Der Winkel der J2-Achse wird automatisch berechnet und als 2. Element festgelegt. Er kann nur gelesen und nicht eingestellt werden. Verglichen mit dem Parameter MEJAR (Verfahrweggrenzen für Gelenkbewegungen) ist eine Begrenzung in einem engeren Bereich möglich.  	0, 0

Tab. 9-2: *Übersicht der Bewegungsparameter (4)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Position des Rückzug-punktes	JSAFE	Reelle Zahl 8	Festlegung der Position des Rückzugspunktes Der Roboter fährt die Position des Rückzugspunktes bei Ausführung des Befehls Mov P_Safe oder bei anliegendem externen SAFEPOS-Signal an. (J1, J2, J3, J4, J5, J6, J7, J8) Einheit: Grad	Modellabhängig
Mechanischer Anschlagpunkt	MORG	Reelle Zahl 8	Festlegung der Position des mechanischen Anschlagpunkts (J1, J2, J3, J4, J5, J6, J7, J8) Einheit: Grad	Modellabhängig
Benutzer-definierter Nullpunkt	USERORG	Reelle Zahl 8	Festlegung des benutzerdefinierten Nullpunkts (J1, J2, J3, J4, J5, J6, J7, J8) Einheit: Grad	Modellabhängig
Fehlermeldung bei Annäherung an den singulären Punkt (siehe auch Abschn. 9.20)	MESNGLSW	Ganze Zahl 1	Freigabe einer Fehlermeldung bei Annäherung an den singulären Punkt. (gesperrt/freigegeben = 0/1) Ist die Fehlermeldung über den Parameter freigegeben, ertönt der Warnton auch dann, wenn der Summer über den Parameter BZR (Summer EIN/AUS) ausgeschaltet ist.	1 (freigegeben)
JOG-Einstellung	JOGJSP	Reelle Zahl 3	Festlegung der Geschwindigkeit für den Gelenk-JOG- und den Schrittbetrieb (Einstellung der Werte H/L, maximaler Übersteuerungswert)	Abhängig vom Mechanismus
	JOGPSP	Reelle Zahl 3	Festlegung der Geschwindigkeit für den Linear-JOG- und den Schrittbetrieb (Einstellung der Werte H/L, maximaler Übersteuerungswert) Der maximale Wert von 250 mm/s kann nicht überschritten werden.	Abhängig vom Mechanismus
Geschwindigkeitsbegrenzung für den JOG-Betrieb	JOGSPMX	Reelle Zahl 1	Geschwindigkeitsbegrenzung im TEACH-Modus Einheit: mm/s (max. 250 mm/s)	250.0

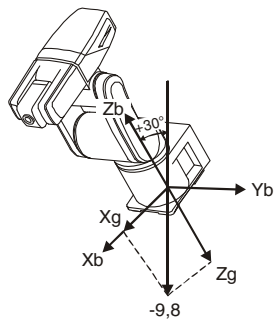
Tab. 9-2: *Übersicht der Bewegungsparameter (5)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Werkstück-koordinaten Funktion steht in folgenden Software-Versionen zur Verfügung: TB: ab Version 1.3 SD-Serie: ab Version P8 SQ-Serie: ab Version N8	WK1CORD : WK8CORD	Reelle Zahl 6	Einstellung der Werkstückkoordinaten für den Werkstück-JOG-Betrieb (X, Y, Z, A, B, C) Einheit: mm oder Grad Die Werkstückkoordinaten werden als Standardkoordinatendaten und als Werkstückkoordinatendaten im JOG-Betrieb verwendet. Bei der Verwendung als Werkstückkoordinatendaten können sich die gültigen Elemente je nach Robotermodell unterscheiden (siehe auch Abschn. 9.8). Die Werkstückkoordinaten werden über die Teaching Box eingestellt. Allerdings können die Koordinatenwerte für diesen Parameter auch direkt eingegeben werden. In diesem Fall wird aber jeder Koordinatenwert der drei Teaching-Punkte, welche die Werkstückkoordinaten definieren, auf 0 gesetzt. (Parameter: WKnWO, WKnWX, WKnWY (n = 1–8))  HINWEIS: Teachen Sie die Werkstückkoordinaten für eine einfache Handhabung in einem Zustand, in dem die Basiskoordinaten nicht konvertiert werden. (Basis- und Weltkoordinatensystem stimmen überein.) Dies ist insbesondere dann wichtig, wenn zwei oder mehr Werkstückkoordinaten definiert werden.	(0.00, 0.00, 0.00 0.00, 0.00, 0.00)

Tab. 9-2: *Übersicht der Bewegungsparameter (6)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Werkstück-koordinaten Funktion steht in folgenden Software-Versionen zur Verfügung: TB: ab Version 1.3 SD-Serie: ab Version P8 SQ-Serie: ab Version N8	WK1WO : WK8WO	Reelle Zahl 3	Einstellung der Position des Koordinatenursprungs der Werkstückkoordinaten als Teaching-Position. (Entspricht der Teaching-Box-Funktion „WO“. Siehe obige Abbildung) (X, Y, Z) Einheit: mm HINWEIS: Auch wenn dieser Koordinatenwert eingegeben wurde, sind die Werkstückkoordinaten nicht definiert. Definieren Sie diese über den Betrieb mit der Teaching-Box.	(0.0, 0.0, 0.0)
	WK1WX : WK8WX	Reelle Zahl 3	Einstellung der „+X“-Achsenposition der Werkstückkoordinaten als Teaching-Position. (Entspricht der Teaching-Box-Funktion „WX“. Siehe obige Abbildung) (X, Y, Z) Einheit: mm HINWEIS: Auch wenn dieser Koordinatenwert eingegeben wurde, sind die Werkstückkoordinaten nicht definiert. Definieren Sie diese über den Betrieb mit der Teaching-Box.	(0.0, 0.0, 0.0)
	WK1WY : WK8WY	Reelle Zahl 3	Einstellung der Position seitlich zur „+Y“-Achsen auf der XY-Ebene der Werkstückkoordinaten als Teaching-Position. (Entspricht der Teaching-Box-Funktion „WY“. Siehe obige Abbildung) (X, Y, Z) Einheit: mm HINWEIS: Auch wenn dieser Koordinatenwert eingegeben wurde, sind die Werkstückkoordinaten nicht definiert. Definieren Sie diese über den Betrieb mit der Teaching-Box.	(0.0, 0.0, 0.0)
Automatische Rückkehr nach einem Interrupt (Siehe auch Abschn. 9.11))	RETPATH	Ganze Zahl 1	Bewirkt die Fortsetzung des Programms nach Auftreten eines Interrupts von der Interruptposition aus Ist die Funktion deaktiviert und der Roboter wird nach einem Interrupt im JOG-Betrieb zu einer anderen Position bewegt, erfolgt die Weiterführung der Roboterbewegung bei Fortsetzung des Programms von der aktuellen Position aus. Der Roboter kehrt nicht zu der Position zurück, an der er sich zum Zeitpunkt des Interrupts befunden hat. 0: Funktion deaktiviert 1: Rückkehr mittels Gelenk-Interpolation 2: Rückkehr mittels Linear-Interpolation HINWEISE: Führen Sie bei Rückkehr mittels Linear-Interpolation kürzere Kreisbewegungen über die 3-Achsen-XYZ-Interpolation aus. Für die Kreis-Interpolation (Mvc, Mvr, Mvr2, Mvr3) kann die Funktion ab Software-Version H4 genutzt werden. Bei der Kreis- und Bogen-Interpolation (Mva) ist die Funktion bei der Einstellung „0“ dieselbe, wie bei der Einstellung „1“.	1 (aktiviert)

Tab. 9-2: *Übersicht der Bewegungsparameter (7)*

<table>
<tr><th colspan="2">Parameter</th><th>Anzahl der Felder/ Zeichen</th><th>Beschreibung</th><th>Werkseinstellung</th></tr>
<tr><td>Montage-richtung</td><td>MEGDIR</td><td>Reelle Zahl
4</td><td>Legt die Richtung und Größe der auf den Roboter wirkenden Erdbeschleunigung in Abhängigkeit von der Montagerichtung für die X-, Y- und Z-Achse des Basiskoordinaten-systems fest (Einheit: mm/s^2). Der Parameter besteht aus 4 Elementen: Montagerichtung, Erdbeschleunigung in X-, Y- und in Z-Richtung
<table>
<tr><th>Montage-richtung</th><th>Montagerichtung, Erdbeschleunigung in X-, Y- und in Z-Richtung</th></tr>
<tr><td>Boden-montage</td><td>(0.0, 0.0, 0.0, 0.0)</td></tr>
<tr><td>Wand-montage</td><td>(1.0, 0.0, 0.0, 0.0)</td></tr>
<tr><td>Decken-montage</td><td>(2.0, 0.0, 0.0, 0.0)</td></tr>
<tr><td>Spezielle Montage ①</td><td>(3.0, ***, ***, ***)</td></tr>
</table>
①Die Zeichen „***“ entsprechen einem numerischen Wert
(siehe folgendes Beispiel)
Beispiel:
Der Roboter wird 30° nach vorne geneigt montiert. Für die Erdbeschleunigung in X-Richtung ergibt sich:
$X_g = 9{,}8 \sin 30° = 4{,}9$
Für die Erdbeschleunigung in Z-Richtung ergibt sich:
$Z_g = 9{,}8 \cos 30° = 8{,}5$
Der Wert muss auf –8,5 gesetzt werden, da der Vektor in die negative Richtung der Z-Achse zeigt.
Für die Erdbeschleunigung in Y-Richtung ergibt sich:
$Y_g = 0{,}0$
Daraus ergibt sich die Einstellung des Parameters: (3.0, 4.9, 0.0, –8.5)
 
</td><td>0.0, 0.0, 0.0, 0.0</td></tr>
</table>

Tab. 9-2: *Übersicht der Bewegungsparameter (8)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Initialisierungsstatus der HAND (siehe auch Abschn. 9.14)	HANDINIT	Ganze Zahl 8	Festlegung der Ausgänge der Schnittstellenkarte für die pneumatische Greifhand nach Einschalten der Spannungsversorgung Der Parameter legt den Handgreiferzustand nach der Initialisierung über die 900er Signale fest. Soll die Einstellung des Handgreiferzustandes nach der Initialisierung über allgemeine E/A (andere als die 900er Signale) (im Parameter HANDTYPE sind andere Signale als die 900er festgelegt) erfolgen, verwenden Sie zur Festlegung der Zustände nicht den Parameter HANDINIT, sondern die Parameter ORSTn. Die über die Parameter ORSTn gesetzten Ausgangsbitmuster entsprechen dem Initialisierungszustand der Signale nach dem Einschalten der Spannungsversorgung.	1,0,1,0,1,0,1,0
Handausführung (siehe auch Abschn. 9.13)	HANDTYPE	Zeichenkette 8	Festlegung der Handausführung (Einfach-/ Doppelmagnetspule = S/D) und Signalnummer Geben Sie erst den Handtyp, dann die Signalnummer an: z. B. D900. Bei einer Einstellung von D900 werden die Signale Nr. 900 und 901 ausgegeben. Ist über die Einstellung „D“ eine Doppelmagnetspule gewählt, setzen Sie die Signalnummern so, dass keine Überlappung der Signale auftritt.	D900, D902, D904, D906, , , ,

Tab. 9-2: *Übersicht der Bewegungsparameter (9)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Hand- und Werkstückbedingungen (Bei optimaler Beschleunigung/Abbremsung und aktivierter Kollisionsüberwachung) (siehe auch Abschn. 9.19.1)			Es werden benutzerdefinierte Bereiche (maximal 32) festgelegt. Wenn der Roboter in diese Bereiche eindringt, wird das als Verfahrwegüberschreitung definiert und als Reaktion darauf kann ein korrespondierendes Signal geschaltet werden.	
	HNDDAT0	Reelle Zahl 7	Einstellung der Startbedingung für die Hand (Festlegung im Werkzeugkoordinatensystem) Nach dem Einschalten der Spannungsversorgung ist dieser Wert wirksam. Der Parameter muss vor Aktivierung der Kollisionsüberwachung im JOG-Betrieb eingestellt werden. Ist der Parameter nicht eingestellt, kann ein fehlerhaftes Ansprechen der Kollisionsüberwachung erfolgen. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Für die Robotermodelle RV-2SDB/3SDB/3SDJB: 3.50, 284.00, 284.00, 286.00, 0.00, 0.00, 75.00 RV-6SD/6SDL: 6.00, 213.00, 213.00, 17.00, 0.00, 0.00, 130.00 RV-12SD/12SDL: 12.00, 265.00, 265.00, 22.00, 0.00, 0.00, 66.00 RH-6SDH: 6.00, 99.00, 99.00, 76.00, 0.00, 0.00, 38.00 RH-12SDH: 12.00, 225.00, 225.00, 30.00, 0.00, 0.00, 15.00 RH-18SDH: 18.00, 258.00, 258.00, 34.00, 0.00, 0.00, 17.00
	HNDDAT1 : HNDDAT8	Reelle Zahl 7	Einstellung der Startbedingung für die Hand (Festlegung im Werkzeugkoordinatensystem) (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Standardlast 0.0,0.0,0.0,0.0,0.0,0.0
	WRKDAT0	Reelle Zahl 7	Einstellung der Startbedingung für das Werkstück (Festlegung im Werkzeugkoordinatensystem) Nach dem Einschalten der Spannungsversorgung ist dieser Wert wirksam. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Für die Robotermodelle RV-S/RH-S: 0.0,0.0,0.0,0.0,0.0,0.0,0.0
	WRKDAT1 : WRKDAT8	Reelle Zahl 7	Einstellung der Werkstückbedingungen (Festlegung im Werkzeugkoordinatensystem) (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	0.0,0.0,0.0,0.0,0.0,0.0,0.0
	HNDHOLD1 : HNDHOLD8	Ganze Zahl 2	Festlegung, ob die Hand beim Befehl HOpen (HClose) geschlossen wird oder nicht (Einstellung für Öffnen, Einstellung für Schließen) (öffnen/schließen = 0/1)	0, 1
Maximale Beschleunigung/ Abbremsung (siehe auch Abschn. 9.19.1)	ACCMODE	Ganze Zahl 1	Festlegung des Startwertes und der optimalen Beschleunigung/Verzögerung (gesperrt/freigegeben = 0/1)	Für die Robotermodelle RH-SDH/RV-SD ……1 Für alle anderen Robotermodelle ……0

Tab. 9-2: *Übersicht der Bewegungsparameter (10)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Einstellzeit für die optimale Beschleunigungs-/ Abbremszeit	JADL	Reelle Zahl 8	Festlegung des Startwertes (nach Einschalten der Spannungsversorgung) der Einstellzeit für die optimale Beschleunigungs-/ Abbremszeit beim Betrieb mit optimaler Beschleunigung/Abbremsung. Dieser Wert wird dem berechneten Wert für die optimale Beschleunigung/Abbremsung überlagert. Ein großer Einstellwert führt bei den Robotermodellen RV-SD zu einer Verkürzung der Zykluszeiten. Gleichzeitig steigt jedoch die Wahrscheinlichkeit von Überlast- und Überhitzungsfehlern. Verkleinern Sie in solchen Fällen die Einstellwerte. Werkseitig sind die Werte so eingestellt, dass Überlast- und Überhitzungsfehler vermieden werden. Die Werte sind für die Beschleunigungs- und die Abbremszeit wirksam. • Definition eines Überlastfehlers: Ein Überlastfehler tritt dann auf, wenn die Größe der Last bei einer Verfahrbewegung mit hoher Geschwindigkeit eine Überhitzung des Motors zur Folge haben könnte. • Definition eines Überhitzungsfehlers: Ein Überhitzungsfehler tritt dann auf, wenn der Encoder aufgrund der hohen Temperatur beschädigt werden könnte. Einheit: % HINWEIS: Dieser Parameter ist nur für die Robotermodelle RV-SD aktiviert.	Für die Robotermodelle RV-2SDB/3SDB/3SDJB 100,100,100,100,100, 100,100,100(%) RV-6SD/12SD 50,50,50,50,50,50,50, 50(%) RV-6SDL/12SDL 35,35,35,35,35,35,35, 35(%)
Optimale Beschleunigung/Abbremsung Nur für die Roboter: RH-6SDH/ RH-6SQH/ RH-12SDH/ RH-12SQH/ RH-20SDH/ RH-20SQH	MAPMODE	Reelle Zahl 1	Wählen Sie bei den Robotern RH-6SDH/ 12SDH/20SDH und RH-6SQH/12SQH/20SQH zwischen der standardmäßigen Beschleunigung/Abbremsung oder der (in Abhängigkeit von der Spindelhöhe der J3-Achse) optimierten Beschleunigung/Abbremsung. 0: standardmäßige Beschleunigung/ Abbremsung (Werkseinstellung) 1: optimierte Beschleunigung/Abbremsung In der Werkseinstellung ist die standardmäßge Beschleunigung/Abbremsung voreinstellt und Vibrationen werden auf ein Minimum reduziert. Sind diese Vibrationen für den Roboterbetrieb nicht von Bedeutung, wählen Sie die optimierte Beschleunigung/ Abbremsung, so dass der Roboter mit hoher Geschwindigkeit arbeiten kann. Weitere Informationen zur Beschleunigung/Abbremsung finden Sie im Technischen Handbuch des Roboters.	0
18- oder 20-kg-Modus Nur für die Roboter: RH-20SDH85xx/ RH-20SQH85xx	LOADMODE	Reelle Zahl 1	Wählen Sie bei den Robotern RH-20SDH85xx und RH-20SQH85xx zwischen den 18- oder 20-kg-Modus. 0: 20-kg-Modus (Werkseinstellung) 1: 18-kg-Modus Umfasst die Last bei eingestellten Bedingungen für 18 kg oder weniger auch das Gewicht der Hand, kann die Zykluszeit bei den Robotern RH-20SDH85xx und RH-20SQH85xx von den üblichen Werten abweichen. In diesem Fall besteht die Möglichkeit, den Roboter mit nahezu den Zykluszeiten der Roboter RH-18SDH85xx und RH-18SQH85xx zu betreiben, indem der Parameter LOADMODE auf „1" gesetzt wird.	0

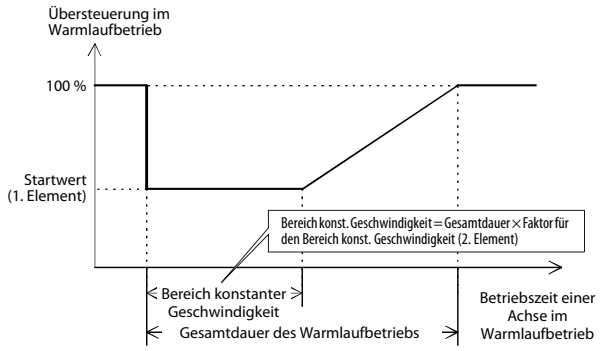
Tab. 9-2: *Übersicht der Bewegungsparameter (11)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Geschwindigkeitsumschaltung Funktion steht in folgenden Software-Versionen zur Verfügung: SD-Serie: ab Version S1w SQ-Serie: ab Version R1w Nur für die Roboter: RH-3SDHR/ RH-3SQHR	SPDOPT	Ganze Zahl 1	Freigabe der Funktion zur Optimierung der Geschwindigkeit direkt nach Einschalten der Spannungsversorgung 0 = gesperrt (Die Funktion zur Optimierung der Geschwindigkeit ist direkt nach dem Einschalten der Spannungsversorgung gesperrt.) 1 = freigegeben (Die Funktion zur Optimierung der Geschwindigkeit ist direkt nach dem Einschalten der Spannungsversorgung freigegeben.) Mit Hilfe des Parameters kann vorgegeben werden, ob die Funktion zur Optimierung der Geschwindigkeit in einem Programm über den SpdOpt-Befehl ein- und ausgeschaltet werden kann.	Für die Robotermodelle RH-3SDHR/3SQHR: 1 Für alle anderen Robotermodelle: 0
Kollisionsüberwachung	COL	Ganze Zahl 3	Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung 1. Element: Kollisionsüberwachung 0 = deaktiviert 1 = aktiviert 2. Element: Aktivierung direkt nach Einschalten der Spannungsversorgung 0 = deaktiviert 1 = aktiviert 3. Element: Freigabe im JOG-Betrieb 0 = deaktiviert 1 = aktiviert 2 = NOERR-Modus Im NOERR-Modus erfolgt keine Fehlerausgabe, auch wenn die Kollisionsüberwachung anspricht. Die Servoversorgung wird jedoch abgeschaltet. Verwenden Sie diesen Modus, wenn kein störungsfreier Betrieb durch häufiges Ansprechen der Kollisionsüberwachung möglich ist. Wenn kein Programmbetrieb (inklusive Anfahren definierter Positionen und schrittweiser Programmausführung) vorliegt, hängt die Festlegung von der Einstellung des JOG-Betriebs (3. Element) ab.	Für die Robotermodelle RV-SD/SQ: 0,0,1 RH-SDH/SQH: 1,0,1 RH-SDHR/SQHR: 1,1,1
Ansprechschwelle der Kollisionsüberwachung	COLLVL	Ganze Zahl 8	Einstellung der Ansprechschwelle im Programmbetrieb für jede Achse Bei Einstellung eines Wertes außerhalb des zulässigen Einstellbereiches wird der nächste zulässige Wert gesetzt. Einstellbereich: 1 bis 500 % Einheit: %	Die Einstellung hängt vom Robotermodell ab.
Ansprechschwelle der Kollisionsüberwachung im JOG-Betrieb	COLLVLJG	Reelle Zahl 8	Einstellung der Ansprechschwelle (Empfindlichkeit) im JOG-Betrieb (inklusive Unterbrechung) für jede Achse. Zur Erhöhung der Ansprechschwelle (Empfindlichkeit) ist der numerische Wert zu verringern. Spricht die Kollisionsüberwachung im JOG-Betrieb auch ohne einen Zusammenstoß an, erhöhen Sie den Wert. Bei Einstellung eines Wertes außerhalb des zulässigen Einstellbereiches wird der nächste zulässige Wert gesetzt. Einstellbereich: 1 bis 500 % Einheit: %	Die Einstellung hängt vom Robotermodell ab.

Tab. 9-2: *Übersicht der Bewegungsparameter (12)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Koordinatensystem für den Handgelenkdrehwinkel (Achse A)	RCD	Ganze Zahl 1	Auswahl des Steuer- und Anzeigemodus des Handgelenkdrehwinkels (Achse A im XYZ-Koordinatensystem) eines 5-achsigen Roboters 2: allgemeiner Winkelmodus Die Achse A wird so gesteuert, dass die Stellung der Hand aufrecht erhalten wird, wenn der Wert der Achse A vor einer Bewegung mit dem Wert nach der Bewegung übereinstimmt. Beachten Sie, dass die Stellung der Hand in Abhängigkeit der Handgelenkneigung (Achse B im XYZ-Koordinatensystem) nicht immer aufrecht erhalten werden kann. Im Normalfall kann die Werkseinstellung beibehalten werden. 0/1/3: allgemeiner Winkelmodus der E-Serie/ Gelenkwinkelmodus/alter allgemeiner Winkelmodus Diese Einstellungen dienen der Kompatibilität von Roboterprogrammen (Positionsdaten) und älteren Robotermodellen (z. B. RV-E3J, RV-E5NJ). Um Programme (Positionsdaten), die für ältere Robotermodelle entworfen wurden, weiterzuverwenden, setzen Sie den Parameter RCD auf den für das ältere Robotermodell festgelegten Wert. Die Modi sind untereinander nicht kompatibel. Bei zwei unterschiedlichen Einstellungen des Parameters können die Handstellungen während einer Roboterbewegung voneinander abweichen, auch wenn die gleichen Positionsdaten angefahren werden. Wählen Sie den Modus so, wie Sie die Positionsdaten bei Ausführung des Programms verwenden möchten.	2 (allgemeiner Winkelmodus)
Freigabe des Warmlaufbetriebs	WUPENA	Ganze Zahl 1	Warmlaufbetrieb freigeben oder sperren 0: gesperrt 1: freigegeben HINWEISE: Bei Einstellung von anderen als den oben genannten Werten ist der Warmlaufbetrieb gesperrt. Bei mehreren Mechanismen muss der Modus für jeden Mechanismus eingestellt werden.	0 (gesperrt)
Auswahl der Achse für den Warmlaufbetrieb	WUPAXIS	Ganze Zahl 1	Auswahl der Achse für den Warmlaufbetrieb durch Ein- und Ausschalten der Bits im Hexadezimalcode (J1, J2 ... mit dem niedrigsten Bit beginnend) Bit EIN: Achse auswählen Bit AUS: Achse nicht auswählen Wählen Sie die Achsen aus, die bei niedrigen Temperaturen große Positionsabweichungen verursachen. HINWEISE: Durch Setzen eines Bits, zu dem keine Achse existiert, wird keine Achse ausgewählt. Wird keine Achse für einen Warmlaufbetrieb festgelegt, bleibt der Warmlaufbetrieb deaktiviert. Bei mehreren Mechanismen müssen die Achsen für jeden Mechanismus ausgewählt werden.	Für die Robotermodelle RV-6SD/12SD 00111000 (J4-, J5- und J6-Achse) RV-3SDB 00001110 RV-3SDJB 00000110 Für alle anderen Robotermodelle: 0

Tab. 9-2: *Übersicht der Bewegungsparameter (13)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Dauer des Warmlaufbetriebs	WUPTIME	Reelle Zahl 2	Einstellung der Zeit für den Warmlaufbetrieb (Gesamtdauer, Wiederholschwelle) in Minuten Gesamtdauer: Legen Sie die Dauer fest, in der die Achse im Warmlaufbetrieb mit reduzierter Geschwindigkeit verfahren wird. (Einstellbereich 0 bis 60) Wiederholschwelle: Legen Sie die Zeit fest, nach der nach Beendigung eines Warmlaufbetriebs und einer kontinuierlichen Betriebspause einer Achse erneut ein Warmlaufbetrieb aktiviert wird (Einstellbereich: 1 bis 1440) HINWEISE: Bei Einstellung eines Wertes außerhalb des Einstellbereichs, wird der nächstliegende Wert innerhalb des Einstellbereichs verwendet. Ist eine Gesamtdauer von „0“ eingestellt, wird der Warmlaufbetrieb deaktiviert. Bei mehreren Mechanismen müssen die Werte für jeden Mechanismus eingestellt werden.	1, 60
Übersteuerung im Warmlaufbetrieb	WUPOvrd	Ganze Zahl 2	Einstellung der Geschwindigkeit im Warmlaufbetrieb (Startwert, Faktor für den Bereich konstanter Geschwindigkeit) in Prozent Startwert: Startwert der Geschwindigkeitsübersteuerung im Warmlaufbetrieb (Einstellbereich 50 bis 100) Faktor für den Bereich konstanter Geschwindigkeit: Stellen Sie den Bereich der konstanten Geschwindigkeit in Bezug zur Gesamtdauer des Warmlaufbetriebs ein (Einstellbereich 0 bis 50) Folgende Abbildung zeigt den Einfluss der eingestellten Werte auf die Geschwindigkeit.	70, 50
				
			HINWEISE: Bei Einstellung eines Wertes außerhalb des Einstellbereichs, wird der nächstliegende Wert innerhalb des Einstellbereichs verwendet. Ist für die Übersteuerung ein Startwert von 100 % eingestellt, bleibt der Warmlaufbetrieb deaktiviert. Bei mehreren Mechanismen müssen die Werte für jeden Mechanismus eingestellt werden.	

Tab. 9-2: *Übersicht der Bewegungsparameter (14)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Freigabe von Fehlermeldungen im Betrieb mit aktivierter Achsenweichheit	CMPERR	Ganze Zahl 1	Mit diesem Parameter kann die Generierung der Fehler 2710 bis 2740, die bei aktivierter Achsenweichheit auftreten können, gesperrt werden. 1: Fehlergenerierung freigeben 0: Fehlergenerierung sperren Die Fehlermeldungen haben folgende Bedeutung: 2710: Die Abweichung von der Sollwertposition ist zu groß. 2720: Der zulässige Gelenkwinkel bei Ausführung des CMP-Befehls wurde überschritten. 2730: Die zulässige Geschwindigkeit bei Ausführung des CMP-Befehls wurde überschritten. 2740: Fehlerhafte Konvertierung von Koordinaten bei Ausführung des CMP-Befehls Tritt einer dieser Fehler auf, liegt eine Fehlfunktion im Betrieb mit aktivierter Achsenweichheit vor. Zur Korrektur des Fehlers ist es notwendig, die geteachten Positionen und den Programminhalt zu überprüfen. Setzen Sie den Parameter nur auf „0“ (Fehlergenerierung sperren), wenn Sie sicher sind, dass durch einen Betrieb ohne die Generierung entsprechender Fehler keine Komplikationen auftreten können.	1 (Fehlergenerierung freigeben)
Anzeigeintervall für Batteriefehler	ITBATERR	Ganze Zahl 1	Festlegung eines Intervalls (in Stunden), nach dem bei entladenen Batterien eine Fehlermeldung erfolgt. Einstellbereich: 1 bis 336 Ein Wert kleiner als „1“ wird als „1“ interpretiert, ein Wert größer als „336“ als „336“.	24
Zeitsynchronisierung mit SPS (nur CRQ)	TIMESYNC	Ganze Zahl 1	Festlegung, ob die Uhr des Roboter-Steuergerätes und die Uhr der SPS synchronisiert werden sollen. synchronisieren/nicht synchronisieren = 1/0	0

Tab. 9-2: *Übersicht der Bewegungsparameter (15)*

9.3 Signalparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung von Signalen.

<table>
<tr><th colspan="2">Parameter</th><th>Anzahl der Felder/ Zeichen</th><th>Beschreibung</th><th>Werkseinstellung</th></tr>
<tr><td>Spezielle E/A-Signale</td><td></td><td></td><td>Eine detaillierte Beschreibung der speziellen E/A-Signale finden Sie in Abschn. 10.3.</td><td></td></tr>
<tr><td>Stopp-Eingangssignal-bearbeitung</td><td>INB</td><td>Ganze Zahl
1</td><td>Definition des Stopp-Eingangs als Standard oder Drahtbrucherkennung (Standard/Drahtbrucherkennung = 0/1)
Das gilt auch für die Eingangssignale STOP und STOP2. Detaillierte Informationen zum SKIP-Eingang finden Sie im Technischen Handbuch des Roboters.</td><td>0 (Standard)</td></tr>
<tr><td>Einlesen der Programm-nummer vom numerischen Eingang bei Eingabe des Startsignals</td><td>PST</td><td>Ganze Zahl
1</td><td>Die Programmauswahl über das normale externe Eingangssignal erfolgt durch Anlegen der Programmnummer als Eingangssignal (IODATA), durch Auswahl des Programms über das PRGSEL-Signal und durch Starten des Programms über das Startsignal.
Durch Freigabe der Funktion über den Parameter PST kann die Programmauswahl über das Signal PRGSEL entfallen. Nach Einschalten des Startsignals wird die Programmnummer über den numerischen Eingang (IODATA) eingelesen.
(gesperrt/freigegeben = 0/1)</td><td>0 (gesperrt)</td></tr>
<tr><td>Ausgabe Roboterfehler</td><td>ROBOTERR</td><td>Ganze Zahl
1</td><td>Die Fehlerklasse wird eingestellt, bei der eine Ausgangsklemme des Steckers EMGOUT zur Fehlerausgabe an der Robotersteuerung aktiviert wird.
Im Auslieferzustand ist der Parameter so eingestellt, dass der Ausgang bei einem Fehler aus jeder Fehlerklasse aktiviert wird.
Beispiel:
Wenn Warnungen ignoriert werden sollen, aber Fehler sowohl aus der leichten, als auch aus der schweren Fehlerklasse berücksichtigt werden sollen, muss der Parameter auf 3 eingestellt werden.
<table>
<tr><th rowspan="2">Einstellung</th><th colspan="3">Fehlerklasse</th></tr>
<tr><th>Warnung</th><th>Leicht</th><th>Schwer</th></tr>
<tr><td>0</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td>1</td><td>—</td><td>—</td><td>●</td></tr>
<tr><td>2</td><td>—</td><td>●</td><td>—</td></tr>
<tr><td>3</td><td>—</td><td>●</td><td>●</td></tr>
<tr><td>4</td><td>●</td><td>—</td><td>—</td></tr>
<tr><td>5</td><td>●</td><td>—</td><td>●</td></tr>
<tr><td>6</td><td>●</td><td>●</td><td>—</td></tr>
<tr><td>7</td><td>●</td><td>●</td><td>●</td></tr>
</table>
●: Ausgangsklemme wird aktiviert
Weitere Informationen zum Stecker EMGOUT finden Sie im Technischen Handbuch des Roboters</td><td>7 (wird bei jeder Fehlerklasse aktiviert)</td></tr>
</table>

Tab. 9-3: *Übersicht der Signalparameter (1)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
CC-Link-Fehler-aufhebung	E7730	Ganze Zahl 1	Ist im Steuergerät eine CC-Link-Schnittstelle installiert, das Steuergerät aber nicht an ein CC-Link-Netzwerk angeschlossen, erfolgt die Fehlermeldung 7730. Das Steuergerät ist nicht mehr betriebsbereit. Der Fehler kann in der Regel nicht zurückgesetzt werden. Mit Hilfe des Parameters ist ein temporäres Zurücksetzen des Fehlers möglich. (Temporäres Zurücksetzen des Fehlers freigeben/sperren = 1/0) Der Parameter ist direkt nach seiner Einstellung über die Teaching Box oder die PC-Software wirksam. Ein Aus- und Wiedereinschalten der Spannungsversorgung ist nicht notwendig. Beim Aus- und Wiedereinschalten der Spannungsversorgung wird der Parameter wieder auf „0" gesetzt (der Fehler kann nicht mehr zurückgesetzt werden), da der eingestellte Wert nicht gespeichert ist.	0 (temporäres Zurücksetzen des Fehlers sperren)
Ausgangsbitmuster beim Rücksetzen (siehe auch Abschn. 9.15)			Ausgangsbitmuster beim Rücksetzen über den Clr-Befehl oder den Eingang OUTRESET Das voreingestellte Ausgangsbitmuster wird auch beim Einschalten der Versorgungsspannung ausgegeben. Wird in Einheiten von 32 Bit mit den folgenden Parametern gesetzt (AUS/EIN/HALTEN = 0/1/*)	
	ORST0	Zeichenkette 4	Setzen der Ausgangsbits 0–31	00000000,00000000, 00000000,00000000
	ORST32 : ORST8016	Zeichenkette 4	Setzen der Ausgangsbits 32–63 : Setzen der Ausgangsbits 8016–8047 Hinweis: Die Ausgangssignale 716 bis 723 sind reserviert. Eine Einstellung der Ausgangssignale 700 bis 715 ist nicht möglich, da diese für systeminterne Zwecke reserviert sind.	00000000,00000000, 00000000,00000000 :
Ausgang beim RESET rücksetzen	SLRSTIO	Ganze Zahl 1	Zurücksetzen der Ausgänge, wenn das Programm zurückgesetzt wird (gesperrt/freigegeben = 0/1)	0
Anzahl der Multi-CPUs (nur bei CRnQ-700)	QMLTCPUN	Ganze Zahl 1	Einstellung der Anzahl CPUs, die in einem Multi-CPU-System installiert sind.	2

Tab. 9-3: *Übersicht der Signalparameter (2)*

<table>
<tr><th colspan="2">Parameter</th><th>Anzahl der Felder/ Zeichen</th><th>Beschreibung</th><th>Werkseinstellung</th></tr>
<tr><td>Highspeed-Kommunikation zwischen Multi-CPUs
(nur bei CRnQ-700)</td><td>QMLTCPU1
:
QMLTCPU4</td><td>Ganze Zahl
4</td><td>Bei Multi-CPU-Systemen muss die Anzahl der Adressen eingestellt werden, mit denen die Datenübertragung zwischen den CPUs Nr. 1 bis 4 über die Highspeed-Kommunikation statt findet.
Es ist darauf zu achten, dass für alle CPUs die Parameter übereinstimmend eingestellt werden. Andernfalls gibt die SPS einen Fehler aus.
Element 1: Größe des freien Anwender-Adressbereichs
(1 k = 1024 Adressen)
Einstellbereich: 1 k–14 k
(Der Maximalwert hängt von der Anzahl der eingesetzen CPUs ab; siehe Tabelle)
<table><tr><th>Anzahl CPUs</th><th>Einstellbereich in Adressen</th></tr><tr><td>2</td><td>0–14 k</td></tr><tr><td>3</td><td>0–13 k</td></tr><tr><td>4</td><td>0–12 k</td></tr></table>Element 2: Anzahl Adressen für automatische Aktualisierung
Einstellbereich: 0–14335
Die Roboter-CPU unterstützt keine automatische Aktualisierung, so dass die Adressanzahl hier auf „0“ eingestellt bleibt.
Element 3: Größe des Systembereichs
Einstellbereich: 1 oder 2
(1024 oder 2048 Adressen)
Element 4: Synchroner Multi-CPU-Start
Einstellbereich:
1: Ja oder 2: Nein
Roboter-CPUs benötigen einige Zeit zum Aufstarten, so dass die Werkseinstellung von „1“ (Synchroner CPU-Start) hier nicht geändert werden sollte.</td><td>1,0,1,1</td></tr>
</table>

Tab. 9-3: *Übersicht der Signalparameter (3)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Eingangsoffset bei Multi-CPUs (nur bei CRnQ-700)	QMLTCPUS	Ganze Zahl 1	Offsetseinstellung der Robotereingangssignale bei Einsatz von Multi-CPUs an der Steuerung CRnQ. Stellen Sie einen Offset von G10000 (Einheit: 1 kWorte) ein und lesen Sie die Daten aus dem festgelegten gemeinsamen Speicher so aus, wie einen Eingang vom Steuergerät. Diese Einstellung erfolgt bei Bedarf, wenn CPUs von anderen iQ-Plattformen (Motion-CPU oder NCCPU) miteinander kombiniert werden und dabei verhindert werden soll, dass sich die von jeder CPU verwendeten gemeinsamen Speicherbereiche überschneiden. Einstellbereich: –1 bis 14 (Ganze Zahl) (–1: Nicht verwendet/ 0 bis 14 kWorte) • Mit der Einstellung –1 wird der Offset automatisch, abhängig vom belegten Steckplatz, eingestellt. (Kompatibel mit früheren Version bis einschließlich N4a) • Bei einer Einstellung von 1 bis 14 wird der Eingangsoffset entsprechend eingestellt Weitere Informationen dazu finden Sie in Abschn. 9.3.1 Wenn mehrere Roboter miteinander verbunden sind und wenn dieser Parameter überall auf den gleichen Wert (gilt NICHT für –1) eingestellt ist, besteht die Möglichkeit, dass von der Ablaufsteuerung an mehrere Roboter gleichzeitig der gleiche Signalstatus eingegeben werden kann. Weitere Informationen zu Multi-CPU-Systemen finden Sie in der Bedienungsanleitung der QCPU (Edition Multi-CPU-Systeme)	–1
Verarbeitungsmodus des Signalausgangs ①	SYNCIO	Ganze Zahl 1	Festlegung des Verarbeitungsmodus der Signalausgabe über M_Out/M_Outb/ M_Out8/M_Outw/M_Out16/M_Out32/ M_Dout und Def IO Kompatibilitätsmodus/Highspeedmodus 1/ Highspeedmodus 2 Kompatibilitätsmodus: Verarbeitung mit herkömmlicher Kompatibilität, ohne eine Beschleunigung des Signal-Erneuerungszyklus Highspeedmodus 1: Beschleunigung bei der Ausgabe der Signale M_Out/M_Outb/ M_Out8/M_Outw/M_Out16/M_Out32 Highspeedmodus 2: Zusätzlich zum Highspeedmodus 1 wird die Ausgabe des Signals M_Dout beschleunigt HINWEIS: Diese Funktion ist für die folgenden Software-Versionen verfügbar: Modellreihe SD: ab Version P7 Modellreihe SQ: ab Version N7	Modellreihe SD: 0 Modellreihe SQ: 1

Tab. 9-3: *Übersicht der Signalparameter (4)*

① Mit diesem Parameter wird eine schnelle Verarbeitung von externen Ausgangssignalen über die Systemstatusvariablen M_Out, usw. ermöglicht. Im folgenden Programmbeispiel wird gezeigt, wie die Verarbeitungsgeschwindigkeit der Ausgangssignale innerhalb der Programmschritte 1 und 4 erhöht werden kann.

Programmbeispiel 1

```
1 M_Out(9) = 1                    'Einschalten des Ausgangssignals 9
2 *ack_check                      'Sprungmarke
3 If M_In(7) = 0 Then *ack_check  'Warten bis Eingangssignal 7 einschaltet
                                  (Warteschleife)
4 M_Out(9) = 0                    'Abschalten des Ausgangssignals 9
5 End                             'Programmende
```

Referenzwert der Geschwindigkeitserhöhung

Mit dem vorherigen Programmbeispiel 1 wird die Verarbeitungszeit um ca. 80 % reduziert. Allerdings sind auch die CC-Link-, Profibus- und Parallel-E/A-Schnittstellen (-karte) mit für die Verarbeitungszeit ausschlaggebend, wenn ständig der Befehl zur Signalausgabe auf 2 oder mehr Kommunikationsverbindungen erteilt wird.

Mit dem folgenden Programmbeispiel 2 wird die Verarbeitungszeit um ca. 75 % reduziert.

Programmbeispiel 2

```
1 M_Out(9) = 1                    'Einschalten des Ausgangssignals 9
2 M_Out8(10) = &H1F               'Ausgabe von &H1F über das Ausgangssignal 10 mit
                                  8-Bit Breite
3 M_Out16(18) = &H3FFF            'Ausgabe von &H3FFF über das Ausgangssignal 18 mit
                                  16-Bit Breite
4 M_Out32(33)=&H7FFFFFFF          'Ausgabe von &H7FFFFFFF über das Ausgangssignal 33
                                  mit 32-Bit Breite
5 End                             'Programmende
```

ACHTUNG:

- ***Unterbrechen Sie immer die Signale, um eine Synchronisation auszulösen. Andernfalls können durch falsche oder unvollständig übertragene Signale Fehlfunktionen auftreten.***
- ***Der Modus der konventionellen Kompatibilität kann genutzt werden, wenn man die gleiche Verarbeitungszeit benötigt, wie bei der normalen (konventionellen) Verarbeitung. Für die Modellreihe SD ist über die Werkseinstellung der konventionelle Modus eingestellt. Es wird im Highspeed-Modus empfohlen, die korrekte Funktion aufgrund der Zykluszeitverkürzung über Signalverriegelungen, wie z. B. Warteschleifen sicherzustellen.***

9.3.1 Eingabe-Offset bei Multi-CPUs (nur für die Steuerung CRnQ)

Fall A

Es wird für den Eingang kein Offset verwendet (Parameter QMLTCPUS = –1)

SPS (Wort-Operand)		Roboter (Bit-Operand)	
Ausgang	U3E0\G10000 bis U3E0\G10511	Eingang	Roboter-CPU Nr. 1 / 10000 bis 18191
	U3E0\G10512 bis U3E0\G11023		Roboter-CPU Nr. 2 / 10000 bis 18191
	U3E0\G11024 bis U3E0\G11535		Roboter-CPU Nr. 3 / 10000 bis 18191
Eingang	U3E1\G10000 bis U3E1\G10511	Ausgang	Roboter-CPU Nr. 1 / 10000 bis 18191
	U3E2\G10000 bis U3E2\G10511		Roboter-CPU Nr. 2 / 10000 bis 18191
	U3E3\G10000 bis U3E3\G10511		Roboter-CPU Nr. 3 / 10000 bis 18191

Tab. 9-4: *Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale*

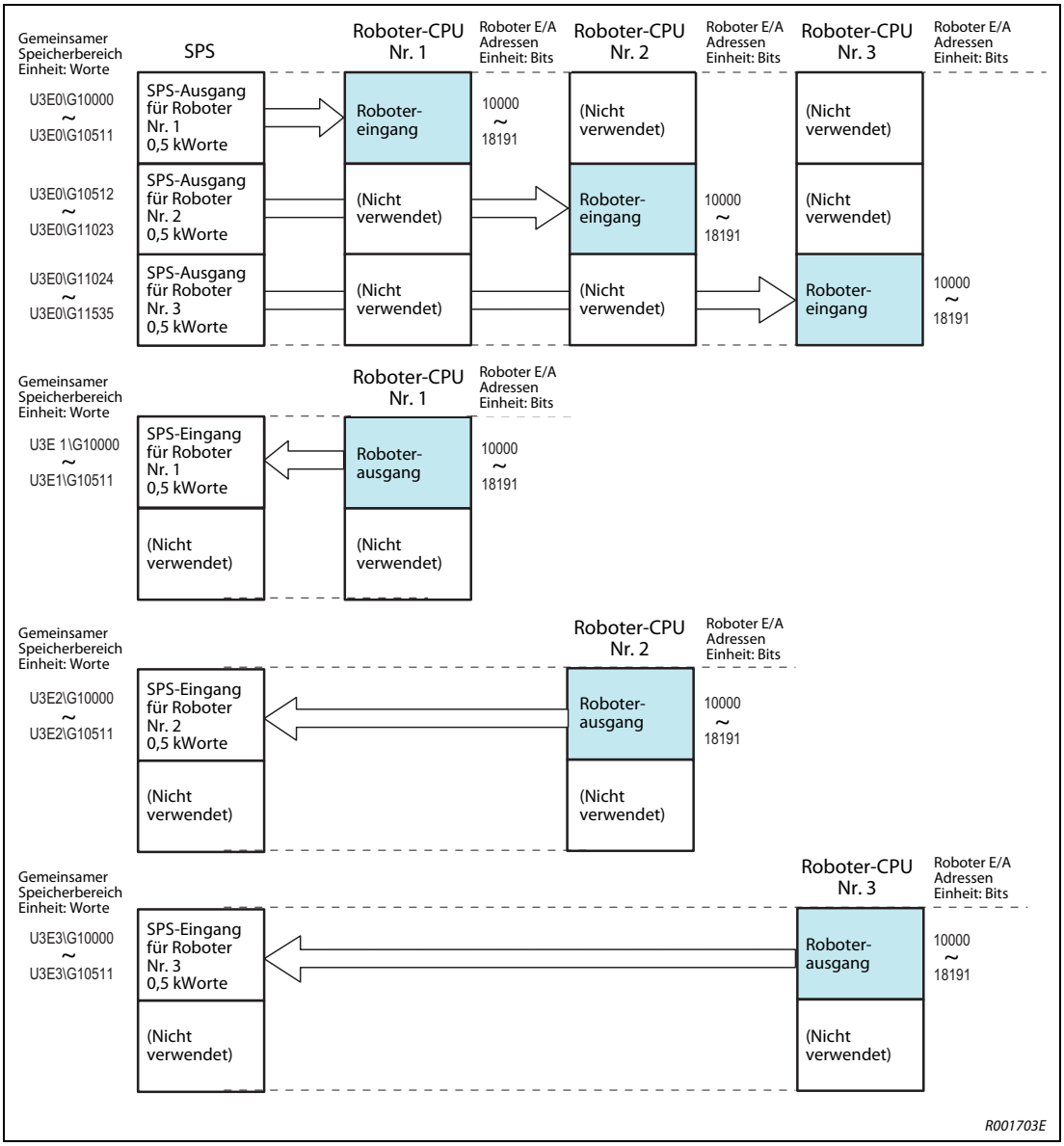


Abb. 9-1: *Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale (Fall A)*

Fall B

Für den Eingang wird ein Offset verwendet (Parameter QMLTCPUS = 0 bis 4)

SPS (Wort-Operand)		Roboter (Bit-Operand)	
Ausgang	U3E0\G10000+① bis U3E0\G10511+①	Eingang	Roboter-CPU Nr. 1 / 10000 bis 18191
	U3E0\G10000+② bis U3E0\G10511+②		Roboter-CPU Nr. 2 / 10000 bis 18191
	U3E0\G10000+③ bis U3E0\G10511+③		Roboter-CPU Nr. 3 / 10000 bis 18191
Eingang	U3E1\G10000 bis U3E1\G10511	Ausgang	Roboter-CPU Nr. 1 / 10000 bis 18191
	U3E2\G10000 bis U3E2\G10511		Roboter-CPU Nr. 2 / 10000 bis 18191
	U3E3\G10000 bis U3E3\G10511		Roboter-CPU Nr. 3 / 10000 bis 18191

Tab. 9-5: *Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale*

① = (Roboter-CPU Nr. 1 QMLTCPUS) × 1024 ② = (Roboter-CPU Nr. 2 QMLTCPUS) × 1024

③ = (Roboter-CPU Nr. 3 QMLTCPUS) × 1024

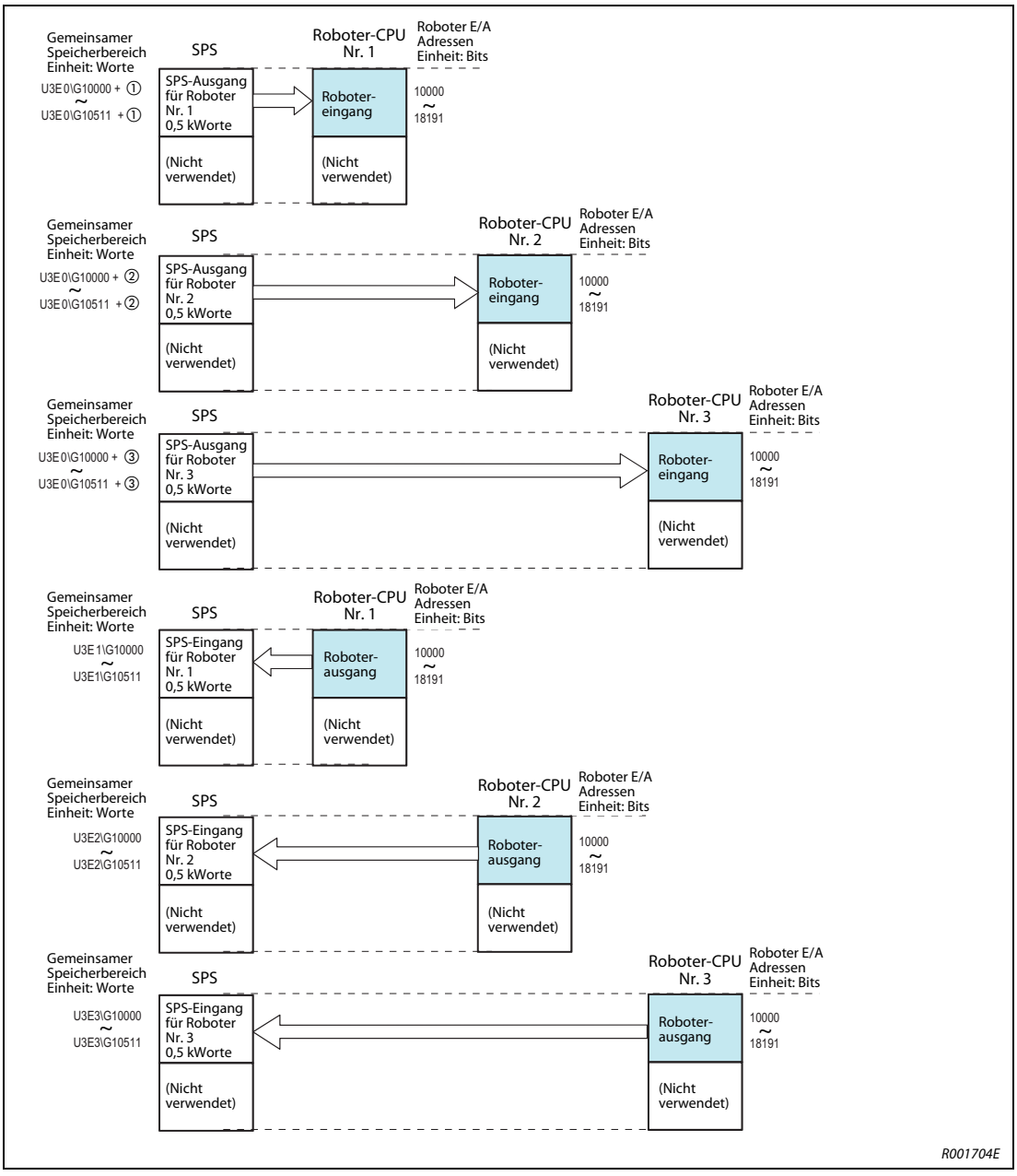


Abb. 9-2: *Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale (Fall B)*

9.4 Betriebsparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung des Steuergeräte- und des Teaching-Box-Betriebs usw.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Summer EIN/AUS	BZR	Ganze Zahl 1	bei einem Fehler EIN oder AUS (AUS/EIN = 0/1)	1 (EIN)
Betriebsrechte zum Rücksetzen	PRSTENA	Ganze Zahl 1	Festlegung, ob zum Rücksetzen des Programms Betriebsrechte erforderlich sind (erforderlich/nicht erforderlich = 0/1) Sind die Betriebsrechte zum Rücksetzen nicht erforderlich, kann das Programm über ein beliebiges externes Gerät zurückgesetzt werden.	0 (erforderlich)
Rücksetzen des Programms über den [MODE]-Schalter des Steuergerätes	MDRST	Ganze Zahl 1	Ein unterbrochenes Programm wird durch Betätigung des [MODE]- Schalters fortgesetzt. (gesperrt/freigegeben = 0/1)	0 (gesperrt)
Anzeige auf dem Steuergerät	OPDISP	Ganze Zahl 1	Festlegung der LED-Anzeige auf dem Steuergerät bei Betätigung des [MODE]-Schalters 0: Anzeige der Geschwindigkeitsübersteuerung 1: aktuelle Anzeige beibehalten	0 (Anzeige der Geschwindigkeitsübersteuerung)
Betriebsrechte zur Programmwahl	OPPSL	Ganze Zahl 1	Festlegung der Betriebsrechte zur Programmwahl für den Automatikbetrieb (OP) (extern/OP = 0/1)	1 (OP)
	RMTPSL	Ganze Zahl 1	Festlegung der Betriebsrechte zur Programmwahl für den Automatikbetrieb (Ext.) (extern/OP = 0/1)	0 (Ext.)
Betriebsrechte der Teaching Box zur Einstellung des Übersteuerungswertes	OVRDTB	Ganze Zahl 1	Festlegung, ob zur Änderung des Übersteuerungswertes über die Teaching Box Betriebsrechte erforderlich sind (nicht erforderlich/erforderlich = 0/1)	RV-2SQ: 1 Alle anderen Roboter: 0
Betriebsartabhängige Geschwindigkeit	OVRDMD	Ganze Zahl 2	Der Übersteuerungswert wird automatisch beim Betriebsartenwechsel geändert. Über das erste Element ist die Übersteuerung festgelegt, wenn die Betriebsart über die Teaching Box geändert wird. Über das zweite Element ist die Übersteuerung festgelegt, wenn die Betriebsart von „AUTOMATIC" auf „MANUAL" geändert wird. Ist die Einstellung „0", wird der aktuelle Wert beibehalten.	0,0
Betriebsrechte zur Einstellung des Übersteuerungswertes	OVRDENA	Ganze Zahl 1	Festlegung, ob zur Änderung des Übersteuerungswertes über das Steuergerät Betriebsrechte erforderlich sind. (erforderlich/nicht erforderlich = 0/1)	1 (erforderlich)
ROM-Modus (siehe auch Abschn. 9.21)	ROMDRV	Ganze Zahl 1	Der Zugriff auf Programme kann zwischen RAM und ROM umgeschaltet werden. 0: RAM-Modus (Standard-Modus) 1: ROM-Modus (Spezial-Modus) 2: Highspeed-RAM-Modus	0 (RAM-Zugriff)

Tab. 9-6: *Übersicht der Betriebsparameter (1)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Kopiert die Daten des RAMs in das ROM (siehe auch Abschn. 9.21)	BACKUP	Zeichenkette 1	Kopiert Programme, Parameter, globale Variablen und Fehler-Logfiles vom RAM in das ROM. Dieser Parameter darf nicht geändert werden.	SRAM → FLROM (nicht einstellbar)
Schreibt die Daten des ROMs in das RAM zurück (siehe auch Abschn. 9.21)	RESTORE	Zeichenkette 1	Schreibt Programme, Parameter, globale Variablen und Fehler-Logfiles vom ROM in das RAM zurück. Dieser Parameter darf nicht geändert werden.	FLROM → SRAM (nicht einstellbar)
Wartungs-intervalle überwachen	MFENA	Ganze Zahl 1	Gibt die Überwachung der Wartungsintervalle frei 1: freigegeben 0: gesperrt Die Funktion steht nur bei den Robotermodellen RV-SD/RH-SDH zur Verfügung. Bei allen anderen Modellen hat die Einstellung dieses Parameters keine Auswirkung.	Für die Robotermodelle RV-SD/RH-SDH: 1, für alle anderen Robotermodelle: 0
Daten-erfassung für Wartungs-intervall	MFINTVL	Ganze Zahl 2	Festlegung der Stufe und der Abtastrate der Datenerfassung zur Bestimmung des Wartungsintervalls 1. Element: Datenerfassungsstufe 1 (niedrigste) bis 5 (höchste) 2. Element: Abtastrate für die Datenerfassung (Einheit: Stunden)	1 (niedrigste), 6 (Stunden)
Wartungsmeldung	MFREPO	Ganze Zahl 2	Festlegung der Methode zur Meldung eines abgelaufenen Wartungsintervalls 1. Element: 1 (Warnmeldung) 0 (keine Warnmeldung) 2. Element: 1 (Ausgabe eines speziellen Signals) 0 (keine Ausgabe eines speziellen Signals)	1 (Warnmeldung) 0 (keine Ausgabe eines speziellen Signals)
Wartungs-daten zurück-setzen Geben Sie vor dem Einlesen dieser Parameter über die Teaching Box alle Parameternamen ein und starten Sie dann den Lesevorgang.	MFGRST	Ganze Zahl 1	Zurücksetzen aller Wartungsdaten, die auf Schmiermittel bezogen sind Erfolgt für eine Achse die Ausgabe der Fehlermeldung im Bereich von 7530, ist das Schmiermittel zu erneuern. Danach sind die auf Schmiermittel bezogenen Wartungsdaten zurückzusetzen. In der Regel werden die Daten über die Programmier-Software zurückgesetzt. Es ist jedoch auch ein Zurücksetzen über die Teaching Box durch Einstellung dieses Parameters möglich.	0: alle Achsen zurücksetzen 1 bis 8: festgelegte Achse zurücksetzen
	MFBRST	Ganze Zahl 1	Zurücksetzen aller Wartungsdaten, die auf die Zahnriemen bezogen sind Erfolgt für eine Achse die Ausgabe der Fehlermeldung im Bereich von 7540, ist der Zahnriemen zu erneuern. Danach sind die auf den Zahnriemen bezogenen Wartungsdaten zurückzusetzen. In der Regel werden die Daten über die Programmier-Software zurückgesetzt. Es ist jedoch auch ein Zurücksetzen über die Teaching Box durch Einstellung dieses Parameters möglich.	0: alle Achsen zurücksetzen 1 bis 8: festgelegte Achse zurücksetzen

Tab. 9-6: *Übersicht der Betriebsparameter (2)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Wiederherstellung von Positionsdaten	DJNT	Reelle Zahl 8	Festlegung der Nullpunktkorrekturdaten für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden. Ein Zugriff auf den Parameter ist nur über ein spezielles Parametermenü in der Programmier-Software möglich.	Modellabhängig
	MEXDTL	Reelle Zahl 6	Festlegung der Standardwerte der Nullpunktkorrekturdaten für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL1	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 1 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL2	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 2 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL3	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 3 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL4	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 4 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDBS	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten der Basis für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0

Tab. 9-6: *Übersicht der Betriebsparameter (3)*

9.5 Befehlsparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Programmausführung.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Anzahl der Programm-plätze	TASKMAX	Ganze Zahl 1	Festlegung der maximalen Anzahl der Programmplätze für eine parallele Ausführung (Multitasking) (Wertebereich 1–32)	8
Programmplatzliste (Einstellung während Multitasking-Betrieb)			Die Einstellung der Betriebsbedingungen für jeden Programmplatz erfolgt während des Multitasking-Betriebs. Diese Einstellungen werden aktiviert, wenn das Programm zurückgesetzt wird	
	SLT1 : SLT32	Zeichenkette 4	Festlegung der Einstellungen (Programmname, Ausführungsformat, Startbedingung und Priorität) jedes Programmplatzes bei der Initialisierung Programmname: Einstellwert Bei Verwendung von Buchstaben sollten nur Großbuchstaben verwendet werden. Kleinbuchstaben werden nicht erkannt. Ausführungsformat: kontinuierlich/zyklisch = REP/CYC REP: Das Programm wird wiederholt ausgeführt. CYC: Das Programm endet nach Ausführung eines Zyklus. (Das Programm endet nicht bei Ausführung einer Endlosschleife über die GOTO-Anweisung.) Startbedingung: Normal/Fehler/Ständig = START/ERROR/ALWAYS START: Die Programmausführung erfolgt über die START-Taste des Steuergerätes oder über das Startsignal ALWAYS: Das Programm wird nach dem Einschalten der Spannungsversorgung ausgeführt. Das Programm hat keinen Einfluss auf die Einschaltroutine. Zur Editierung eines Programmes mit der Startbedingung „ALWAYS“ muss zuerst das ALWAYS-Attribut zurückgesetzt werden. Ändern Sie die Startbedingung „ALWAYS“ auf „START“ und schalten Sie anschließend die Spannungsversorgung aus und wieder ein, um die ständige Ausführung des Programms zu unterbrechen. ERROR: Das Programm wird nach Auftreten eines Fehlers ausgeführt. Das Programm hat keinen Einfluss auf die Einschaltroutine. In Programmen mit der Startbedingung ALWAYS oder ERROR können folgende Befehle nicht ausgeführt werden: Mov, Mvs, MVR, Mvr2, Mvr3, Mvc, Mva, DRIVE, GetM, RelM, JRC Priorität: 1 bis 31 Die Einstellung legt die Anzahl der auszuführenden Zeilen für einen Durchgang fest. Die Funktion entspricht der Priority-Anweisung. Ist die Priorität z. B. für SLT1 auf „1“, für SLT 2 auf „2“ gesetzt, werden nach Abarbeitung von 1 Zeile des Programms in SLT1 2 Zeilen des Programms in SLT2 ausgeführt. Es werden also mehr Programmteile in SLT 2 ausgeführt, d. h. deren Priorität ist höher.	"", REP, START, 1

Tab. 9-7: *Übersicht der Befehlsparameter (1)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Programm-wahl speichern	SLOTON	Ganze Zahl 1	Dieser Parameter legt fest, ob der Programmname bei Auswahl des Programms im Programmplatzparameter SLT1 gespeichert werden soll und ob das Programm nach Beendigung des Zyklus weiterhin ausgewählt bleibt • Speicherung des Programmnamens bei Auswahl des Programms (Bit 0, Speicherung aktiviert/deaktiviert = 1/0) Aktiviert: Bei Auswahl des Programms wird der Programmname im Programmplatzparameter SLT1 gespeichert. Nach Einschalten der Spannungsversorgung wird das im Programmplatzparameter SLT1 gespeicherte Programm ausgewählt. Deaktiviert: Bei Auswahl des Programms wird der Programmname nicht im Programmplatzparameter SLT1 gespeichert. Nach Einschalten der Spannungsversorgung wird das im Programmplatzparameter SLT1 gespeicherte Programm ausgewählt. • Programmaufrechterhaltung nach Beendigung des Zyklusbetriebs (Bit 1, Programm aufrechterhalten/ nicht aufrechterhalten = 1/0) Aufrechterhalten: Das Programm bleibt auch weiterhin ausgewählt, auch wenn im Zyklusbetrieb ein Zyklus beendet wurde. Der Parameter wird nicht auf „P.0000“ gesetzt. Nicht aufrecherhalten: Die Programmauswahl wird zurückgesetzt, wenn das Programm zurückgesetzt wird. Der Parameter wird auf „P.0000“ gesetzt. Einstellwerte und Funktionen: 0: Speicherung deaktiviert/ Programm nicht aufrechterhalten 1: Speicherung aktiviert/ Programm nicht aufrechterhalten (Standardwert) 2: Speicherung deaktiviert/ Programm aufrechterhalten 3: Speicherung aktiviert/ Programm aufrechterhalten	1 (aktiviert)
X- und SERVO-Befehle in Programmplätzen mit der Startbedingung ALWAYS zulassen (siehe auch Abschn. 9.12)	ALWENA	Ganze Zahl 1	Die Befehle XRun, XLoad, XStp, XRst, Servo und Reset Err werden für Programmplätze freigegeben, deren Startbedingung in den Programmplatzparametern SLTn auf „ALWAYS“ gesetzt ist. (freigeben/sperren = 1/0)	0 (gesperrt)
Benutzerbasisprogramm (siehe auch Seite 5-16)	PRGUSR	Zeichenkette 1	Ein Benutzerbasisprogramm wird zur Deklaration von benutzerdefinierten externen Variablen verwendet. Die Variablen werden über die Befehle Def oder Dim deklariert. Sollen im Benutzerbasisprogramm Feldvariablen erstellt und als externe Variablen eingesetzt werden, so ist eine zweite Deklaration über die Dim-Anweisung in dem Programm erforderlich, in dem sie verwendet werden. Einzelne Variablen erfordern keine erneute Deklaration.	—

Tab. 9-7: *Übersicht der Befehlsparameter (2)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Programm fortsetzen	CTN	Ganze Zahl 1	Für den Programmplatz 1 wird nach dem Ausschalten der Spannungsversorgung die aktuelle Position innerhalb der Anwendung gespeichert. Nach dem nächsten Einschalten der Spannungsversorgung startet die Anwendung von dieser gespeicherten Position. Gespeichert wird die Übersteuerung, die Programmzeile, Programmvariable und der Zustand des Ausgangssignals. (freigegeben/gesperrt = 1/0) Beachten Sie die folgenden Hinweise bei Verwendung der Funktion. Bei Nichtbeachtung kann das Programm zerstört werden. • Bei Robotern, die über Achsen ohne Bremse verfügen, kann es zu einem Heruntersinken des Armes aufgrund der Erdanziehungskraft oder zu Rotationen kommen, wenn die Spannungsversorgung ausgeschaltet wird. Treffen Sie daher geeignete Vorsichtmaßnahmen (z. B. Unterstützen des Roboterarms), wenn Sie die Funktion verwenden. • Nur das Programm im Programmplatz 1 kann aus dem „Standby-Modus" gestartet werden. Die Programme aus den anderen Programmplätzen werden aus dem „Reset-Modus" gestartet. • Die Parameter SLTn, SLOTON und TASKMAX können nach Freigabe der Funktion nicht mehr verändert werden. Ändern Sie die Parameter vor einer Freigabe der Funktion. • Änderungen der Programmplatzparameter SLTn nach Freigabe der Funktion werden nicht übernommen. Deaktivieren Sie die Funktion, schalten Sie die Spannungsversorgung aus und wieder ein und ändern Sie dann die Einstellwerte der Parameter.	0 (gesperrt)
JRC-Befehl (Multirotationsfunktion der Achsen)			Festlegung des Ausführungsstatus des JRC-Befehls	
	JRCEXE	Ganze Zahl 1	Freigabe des JRC-Befehls (freigegeben/gesperrt = 1/0)	0 (Ausführung gesperrt)
	JRCQTT	Reelle Zahl 8	Im Parameter JRCQTT werden die Werte festgelegt, um die benutzerdefinierte Achsen verschoben werden. Die Achsen werden in der Reihenfolge J1, J2, J3 bis J8 angegeben. J7 und J8 sind zusätzliche Roboterachsen. Die Einheit der Werte aus JRCQTT ist im Parameter AXUNT festgelegt.	JRC freigegeben: 0,0,0,0,0,360,0,0 oder 0,0,0,360,0,0,0,0 JRC gesperrt: 0,0,0,0,0,0,0,0
	JRCORG	Reelle Zahl 8	Festlegung der Grundposition für JRC = 0 Diese Einstellung ist nur bei benutzerdefinierten Achsen möglich. Die Einheit der Werte aus JRCORG ist im Parameter AXUNT festgelegt.	0,0,0,0,0,0,0,0
Einstellung der Zusatzachsen	AXUNT	Ganze Zahl 16	Festlegung der Einheit (Winkel (°)/Länge (mm) = 0/1)	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

Tab. 9-7: *Übersicht der Befehlsparameter (3)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Benutzerdefinierter Fehler	UER1 : UER20	Ganze Zahl 1, Zeichenkette 3	Legt eine Fehlermeldung, die Fehlerursache und die Fehlerbehebung für einen benutzerdefinierten Fehler fest. Es können maximal 20 benutzerdefinierte Fehler festgelegt werden. Das erste Element legt eine Fehlernummer zwischen 9000 und 9299 fest. Der Standardwert 9900 kann nicht festgelegt werden. Im zweiten Element wird die Fehlermeldung definiert. Im dritten Element wird die Fehlerursache definiert. Im vierten Element wird die Fehlerbehebung definiert. Enthält die Fehlermeldung ein Leerzeichen, ist die ganze Meldung in Anführungszeichen zu setzen (""). Beispiel: 9000,"Zeitüberschreitung", "Kein Signal","Taste betätigen"	9900,"Fehlermeldung", "Fehlerursache", "Fehlerbehebung"
Einheit für drehende Positionselemente	PRGMDEG	Ganze Zahl 1	Festlegung der Einheit für drehende Elemente von Positionsdaten 0: RAD 1: Grad Beispiel: M1 = P1.A Die Einheit ist definiert. Für den direkten Zugriff auf Positionskomponenten ist die Standardeinheit Radiant. Die Standardeinheit für Positionskonstanten (P1 = (100,0,300,0,180,0,180)(7,0)) ist Grad. Die Einstellung dieses Parameters ist hierbei unwirksam.	0 (RAD)
Programmiermethode	RLNG	Ganze Zahl 1	Auswahl der Programmiermethode 2: MELFA-BASIC V 1: MELFA-BASIC IV	1
Landessprache	LNG	Zeichenkette 1	Auswahl der angezeigten Landessprache JPN = Japanisch ENG = Englisch Der Parameter beeinflusst folgende Funktionen: • Sprache auf der LCD-Anzeige der Teaching Box • Fehlermeldungen, die über Datenkommunikation augelesen werden (Standardschnittstelle RS232C, zusätzliche serielle Schnittstelle, Ethernet-Schnittstelle)	Die japanische Version wird durch die Zeichenkette "JPN", die englische durch die Zeichenkette "ENG" angezeigt.
Erweiterung von externen Variablen	PRGGBL	—	Zur Nutzung des erweiterten Bereichs ist Parameter PRGGBL von „0 (Standard/Grundeinstellung)“ auf „1 (erweiterter Bereich)“ zu ändern und die Spannungsversorgung aus und wieder einzuschalten. Wurde für programmexterne und benutzerdefinierte externe Variablen zweimal derselbe Name vergeben, erfolgt beim Einschalten der Spannungsversorgung eine Fehlermeldung und eine Erweiterung des Bereichs ist nicht möglich. Ändern Sie in diesem Fall die Namen der benutzerdefinierten Variablen.	0

Tab. 9-7: *Übersicht der Befehlsparameter (4)*

9.6 Kommunikationsparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Kommunikation.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Kommunikationseinstellungen			Die Kommunikationseinstellungen beziehen sich auf die RS232C- und Ethernet-Schnittstelle am Steuergerät. Da die RS232C-Schnittstelle in Kombination mit der Roboter-Programmier-Software verwendet wird, entfällt hier in der Regel eine Einstellung. Weitere Informationen zur Kommunikationseinstellung der RS232C-Schnittstelle finden Sie in Abschn. 9.16 und der Ethernet-Schnittstelle in Abschn. 9.17.	
	COMDEV	Zeichenkette 8	Der Parameter dient bei Verwendung des Open-Befehls in MELFA-BASIC V der Zuweisung der Kommunikationsleitungen an die Schnittstellen COM1 und COM2. Bei Verwendung der Data-link-Methode in einer OPEN-Anweisung muss der Parameter zuvor eingestellt werden. Der Parameter legt also fest, welches Gerät über welche Kommunikationsleitung COMn ($1 \leq n \leq 8$) angesprochen wird. Die einzelnen Elemente sind von links beginnend wie folgt angeordnet: COM1, COM2,…, COM8. Soll die Data-link-Methode über Ethernet genutzt werden, müssen OPT11 bis OPT19 entsprechend zugeordnet werden. In diesem Fall ist der Standardschnittstelle RS232C des Steuergerätes bereits COM1 zugewiesen. HINWEIS: Dieser Parameter kann für Roboter-CPUs der Modellreihe CRnQ-700 und deren Antriebseinheit nicht verwendet werden, da hier die Kommunikationsschnittstelle dafür nicht vorbereitet ist.	"RS232", , , , , , ,
Für RS232	CBAU232	Ganze Zahl 1	Festlegung der Übertragungsrate (9600, 19200)	9600
	CPRTY232	Ganze Zahl 1	Festlegung der Parität (0: keine, 1: ungerade, 2: gerade)	2 (gerade)
	CSTOP232	Ganze Zahl 1	Festlegung des Stoppbits (1, 2)	2 (Stoppbit)
	CTERM232	Ganze Zahl 1	Festlegung des Endezeichens (0: CR, 1: CR + LF)	0 (CR)
	CPRC232	Ganze Zahl 1	Kommunikationsart (Protokoll) 0: für Software RT ToolBox2 (kein Protokoll) Wird mit dieser Einstellung eine Datenübertragung ausgeführt (Befehle Open, Print und Input), muss das externe Gerät die Zeichen „PRN“ an den Anfang der zu übertragenen Daten setzen. 1: für Software RT ToolBox2 (Protokoll) Die Einstellung muss auch PC-seitig erfolgen. 2: für die Data-link-Methode mit dem Roboterprogramm Bei dieser Einstellung ist keine Verbindung mit der Software RT ToolBox2 möglich.	1

Tab. 9-8: *Übersicht der Kommunikationsparameter (1)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
RT Tool Box2 Einstellung der Kommunikationsart	COMSPEC	Ganze Zahl 1	Einstellung der Kommunikationsart zwischen Robotersteuerung und der Software RT ToolBox2 0: Konventionelle Kommunikation 1: Hochzuverlässige Kommunikation Im Vergleich zur konventionellen Kommunikation ist die Übertragungsrate bei der hochzuverlässigen Kommunikation gering. Diese langsame Kommunikationsart kommt dann zum Einsatz, wenn in der Umgebung der externen Geräten mit hohen Störeinflüssen zu rechnen ist. HINWEIS: Die hochzuverlässige Kommunikation steht nur bei den folgenden Modellen in der jeweiligen Version zur Verfügung: • Robotersteuerung: Modelle SD: ab Version P8 Modelle SQ: ab Version N8 • RT ToolBox2: ab Version 1.5 Stellt man diese Kommunikationsart bei anderen Modellen ein, erfolgt hier die Kommunikation auf die konventionelle Art	
Für Ethernet	NETIP	Zeichenkette 1	IP-Adresse der Robotersteuerung	SD-Serie: 192.168.0.20 SQ-Serie: 192.168.100.1
	NETMSK	Zeichenkette 1	Subnetzmaske	255.255.255.0
	NETPORT	Numerischer Ausdruck 10	Schnittstellennr. Bereich 0 bis 32767 Für externe Steuerungsfunktionen in Echtzeit Entsprechend der Einstellung des Parameters COMDEV (OPT11), (OPT12), (OPT13), (OPT14), (OPT15), (OPT16), (OPT17), (OPT18), (OPT19)	 10000, 10001, 10002, 10003, 10004, 10005, 10006, 10007, 10008, 10009
	CPRCE11 CPRCE12 CPRCE13 CPRCE14 CPRCE15 CPRCE16 CPRCE17 CPRCE18 CPRCE19	Numerischer Ausdruck 9	Protokoll 0: kein Protokoll 1: Protokoll 2: Data-link-Methode Diese Einstellung ist notwendig, wenn die Data-link-Methode über die Ethernet-Schnittstelle erfolgen soll. Entsprechend der Einstellung des Parameters COMDEV (OPT11), (OPT12), (OPT13), (OPT14), (OPT15), (OPT16), (OPT17), (OPT18), (OPT19)	 0, 0, 0, 0, 0, 0, 0, 0, 0

Tab. 9-8: *Übersicht der Kommunikationsparameter (2)*

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Für Ethernet	NETMODE	Numerischer Ausdruck 9	Server-Zuordnung (1: Server, 0: Client) Diese Einstellung ist notwendig, wenn die Datenverbindung über die Ethernetschnittstelle erfolgen soll. Entsprechend der Einstellung des Parameters COMDEV (OPT11), (OPT12), (OPT13), (OPT14), (OPT15), (OPT16), (OPT17), (OPT18), (OPT19)	 1, 1, 1, 1, 1, 1, 1, 1, 1
	NETHSTIP	Numerischer Ausdruck 9	IP-Adresse des Ziel-Servers für die Datenkommunikation Diese Einstellung ist notwendig, wenn die Datenverbindung über die Ethernetschnittstelle erfolgen soll und sie ist nur dann gültig, wenn der Parameter NETMODE auf „Client" eingestellt wird. Entsprechend der Einstellung des Parameters COMDEV (OPT11), (OPT12), (OPT13), (OPT14), (OPT15), (OPT16), (OPT17), (OPT18), (OPT19)	 192.168.0.2, 192.168.0.3, 192.168.0.4, 192.168.0.5, 192.168.0.6, 192.168.0.7, 192.168.0.8, 192.168.0.9, 192.168.0.10
	MXTTOUT	Numerischer Ausdruck 1 (0 bis 32767)	Wert für die Zeitüberschreitung bei der Ausführung von externen Steuerungsbefehlen in Echtzeit (Vielfaches von 7,1 msec, –1 für Deaktivierung)	–1

Tab. 9-8: *Übersicht der Kommunikationsparameter (3)*

9.7 Standard-Werkzeugkoordinaten

Die Einstellung der Werkzeugdaten ist bei einer Verschiebung des Werkzeugmittelpunkts (Tool Centre Point) erforderlich. Dies ist z. B. nach der Montage eines Handgreifers der Fall. Bei Werkseinstellung sind die Werkzeugkoordinaten auf Null gesetzt, d. h. der Werkzeugmittelpunkt liegt mittig im Handflansch.

Die Einstellung kann auf drei Arten erfolgen:

- Einstellung der Werkzeugdaten über den Parameter MEXTL
- Einstellung der Werkzeugdaten im Roboterprogramm über den Befehl Tool
- Einstellung der Werkzeugnummer über die Variable M_Tool
  Die Werkzeugdaten sind dabei in den Parametern MEXTL1 bis 16 festgelegt (siehe auch Abschn. 7.2.48).

Auf den folgenden Seiten finden Sie Einstellbeispiele für die verschiedenen Robotermodelle.

9.7.1 Aufbau der Werkzeugdaten

Die Werkzeugdaten sind wie folgt aufgebaut: X, Y, Z, A, B, C.

X-, Y-, Z-Achse:	Verschiebung des Werkzeugmittelpunkts bezogen auf den Handflansch im Werkzeugkoordinatensystem
A-Achse:	Drehung um die X-Achse im Werkzeugkoordinatensystem
B-Achse:	Drehung um die Y-Achse im Werkzeugkoordinatensystem
C-Achse:	Drehung um die Z-Achse im Werkzeugkoordinatensystem

Beispiel ▽ Die folgenden Beispiele zeigen die verschiedenen Einstellmethoden an unterschiedlichen Robotermodellen.

5-achsiger vertikaler Knickarmroboter

- Einstellung über Parameter
  Parameter MEXTL: 0,0,95,0,0,0
- Einstellung über Tool-Befehl
  1 Tool (0,0,95,0,0,0)

Beim 5-achsigen Roboter ist aufgrund des Bewegungsbereiches nur die Einstellung der Z-Achsen-Komponente wirksam. Die Einstellung anderer Achsen wird ignoriert.

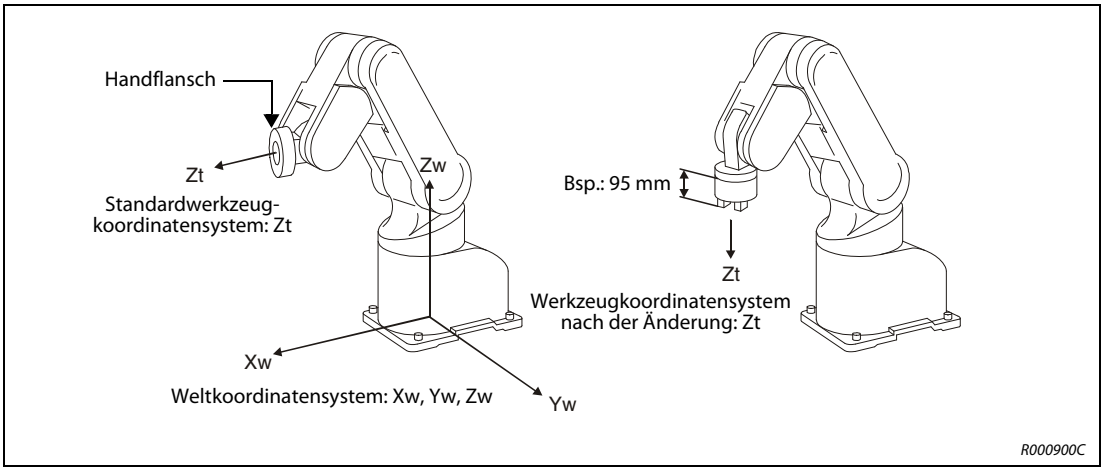


Abb. 9-3: *Werkzeugkoordinaten beim 5-achsigen Knickarmroboter*

6-achsiger vertikaler Knickarmroboter

- Einstellung über Parameter
  Parameter MEXTL: 0,0,95,0,0,0
- Einstellung über Tool-Befehl
  1 Tool (0,0,95,0,0,0)

Beim 6-achsigen Roboter sind innerhalb des Bewegungsbereiches verschiedene Stellungen möglich.

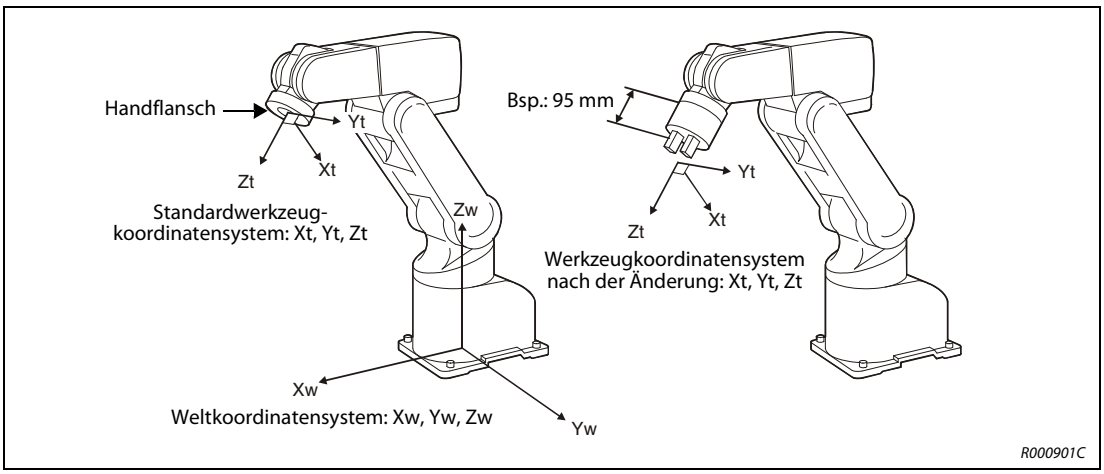


Abb. 9-4: *Werkzeugkoordinaten beim 6-achsigen Knickarmroboter*

4-achsiger SCARA-Roboter

- Einstellung über Parameter
  Parameter MEXTL: 0,0,–10,0,0,0
- Einstellung über Tool-Befehl
  1 Tool (0,0,–10,0,0,0)

Beim 4-achsigen SCARA-Roboter erfolgt die Einstellung des Werkzeugmittelpunktes durch Parallelverschiebung. Die Einstellung der Orientierung des Werkzeugkoordinatensystems unterscheidet sich von der des vertikalen Knickarmroboters.

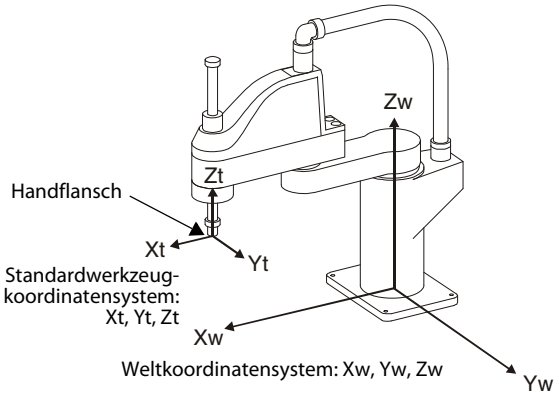


Abb. 9-5: *Werkzeugkoordinaten beim 4-achsigen SCARA-Roboter*

△

Die für die Werkzeug-Konvertierungsdaten relevanten Achsen sind vom Robotermodell abhängig. Folgende Tabelle zeigt den Zusammenhang:

Robotermodell	Anzahl der Achsen	Relevante Achsen					
		X	Y	Z	A	B	C
RV-2SQB/2SDB RV-3SQB/3SDB RV-6SQ/6SQL/6SD/6SDL RV-12SQ/12SQL/12SD/12SDL	6	✔					
RV-3SQJB/3SDJB	5	0		✔	0		
RH-6SDH/12SDH/18SDH/20SDH	4	✔			0		✔

Tab. 9-9: *Werkzeug-Konvertierungsdaten und relevante Achsen*

✔: für die Werkzeug-Konvertierungsdaten gültige Achse
0: der Wert ist auf „0“ festgelegt (Eine andere Einstellung kann sich nachteilig auf den Betrieb auswirken.)

9.8 Standard-Basiskoordinaten

Die Position des Weltkoordinatensystems ist in der Werkseinstellung auf „0“ gesetzt. Dass heißt, das Basiskoordinatensystem (Installationsort des Roboters) ist deckungsgleich mit dem Weltkoordinatensystem (Bezug für die aktuelle Positions des Roboters).
Mit Hilfe der Basiskonvertierung kann der Nullpunkt des Weltkoordinatensystems so verschoben werden, dass er nicht mehr im Mittelpunkt der J1-Achse liegt.
Die Basiskonvertierung bewirkt eine Änderung der Lage von Weltkoordinatensystem und Basiskoordinatensystem zueinander. Führt der Roboter daher den Befehl aus, sich zu einer Positon zu bewegen, die vor der Konvertierung geteacht wurde, fährt er nach der Konvertierung eine andere als die gewünschte Position an. Damit Sie die Basiskonvertierung sinnvoll nutzen können, müssen Sie verstanden haben, welchen Einfluss die Funktion auf vorher geteachte Positionen hat.
Es stehen drei Möglichkeiten zur Festlegung des Weltkoordinatensystems zur Verfügung:

- Verschiebung des Weltkoordinatensystems über die Basiskonvertierungsdaten mit dem Parameter MEXBS
- Verschiebung des Weltkoordinatensystems über die Nummer eines Basiskoordinatenssystems mit dem Parameter MEXBSNO
- Verschiebung des Weltkoordinatensystems im Roboterprogramm über den Befehl Base

9.8.1 Aufbau der Basiskoordinatendaten

Die Basiskoordinatendaten sind wie folgt aufgebaut: X, Y, Z, A, B, C.

X-, Y-, Z-Achse:	Verschiebung des Basiskoordinatensystems bezogen auf das Weltkoordinatensystem
A-Achse:	Drehung um die X-Achse im Weltkoordinatensystem
B-Achse:	Drehung um die Y-Achse im Weltkoordinatensystem
C-Achse:	Drehung um die Z-Achse im Weltkoordinatensystem

Beispiel ▽

- Einstellung über Parameter
  Parameter MEXBS: 100,150,0,0,0,–30
- Einstellung über Base-Befehl
  1 Base (100,150,0,0,0,–30)

Eine Änderung des Basiskoordinatensystems ist im Normalfall nicht nötig. Beachten Sie, dass die Änderung des Basiskoordinatensystems über den Base-Befehl innerhalb eines Programms zu unvorhersehbaren Bewegungen führen kann.

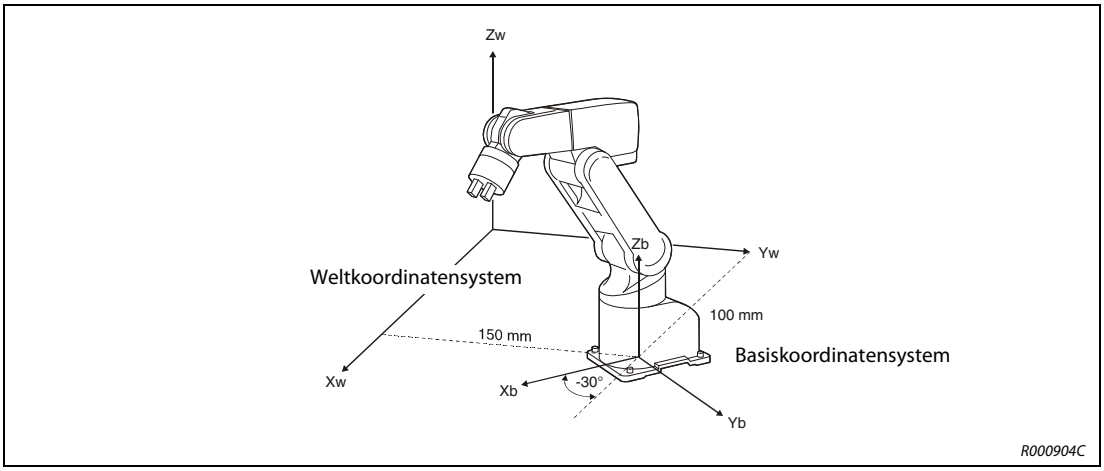


Abb. 9-6: *Basiskoordinatensystem*

△

Die für die Standard-Basiskonvertierungsdaten relevanten Achsen sind vom Robotermodell abhängig. Folgende Tabelle zeigt den Zusammenhang:

Robotermodell	Anzahl der Achsen	Relevante Achsen					
		X	Y	Z	A	B	C
RV-2SQB/2SDB RV-3SQB/3SDB RV-6SQ/6SQL/6SD/6SDL RV-12SQ/12SQL/12SD/SDL	6	✔					
RV-3SQJB/3SDJB	5		✔		0		
RH-3SDHR/3SDQR RH-6SDH/12SDH/18SDH/20SDH	4		✔		0		✔

Tab. 9-10: *Basis-Konvertierungsdaten und relevante Achsen*

✔: für die Basis-Konvertierungsdaten gültige Achse
0: der Wert ist auf „0“ festgelegt (Eine andere Einstellung kann sich nachteilig auf den Betrieb auswirken.)

GEFAHR:

Da sich durch die Basis-Konvertierung der Bezugspunkt der aktuellen Position verändert, sind die bis dahin geteachten Daten nicht weiterverwendbar. Beim versehentlichen Anfahren von Daten, die vor der Konvertierung geteacht worden sind, kann sich der Roboter zu unvorhersehbaren Positionen bewegen. Dabei können Sach- und Personenschäden auftreten.

Prüfen Sie vor der Ausführung der Funktion die Beziehung zwischen den ursprünglich geteachten Positionen und den Positionen, die nach der Konvertierung angefahren werden, um einen einwandfreien Betrieb des Roboters zu gewährleisten.

9.9 Benutzerdefinierter Bereich

Die Funktion des benutzerdefinierten Bereichs dient zur ständigen Überwachung, wann der Roboter mit seinem TCP in einen über Parameter definierten Bereich eindringt. Der Anwender kann wählen, ob ein Statussignal ausgegeben oder ein fehlerbedingter Stopp ausgeführt wird, wenn sich der Roboter innerhalb des definierten Bereiches befindet. Dieser Status kann über die E/A-Schnittstelle oder die Statusvariablen ausgegeben werden.

Mit diese grundlegenden Funktion können Kollisionen vermieden werden, wenn der Roboter mit anderen Maschinen zusammenarbeitet, die sich innerhalb gemeinsamer Arbeitsbereiche bewegen.

Neben dem Arbeitsbereich kann mit dieser Funktion auch beurteilt werden, wo sich der Roboter in Bezug auf den Stellungsdatenbereich oder den Bereich von Zusatzachsen aufhält.

Über die folgenden Parametereinstellungen sind die benutzerdefinierten Bereiche verwendbar.

- Parameter AREAnCS:
  Auswahl des Koordinatensystems, welches das Referenzsystem darstellt
- Parameter AREAnP1 und AREAnP2:
  Festlegung des benutzerdefinierten Bereichs
- Parameter AREAnME:
  Festlegung des zu überwachenden Mechanismus
- Parameter AREAnAT:
  Festlegung der Verhaltens, wenn der Roboter in den benutzerdefinierten Bereich eindringt

9.9.1 Auswahl des Koordinatensystems

Wenn der Betrieb nach Änderung des Basiskoordinatensystems über den Base-Befehl o. Ä. fortgesetzt wird, hat der Anwender die Möglichkeit, den benutzerdefinierten Bereich entweder gleichmäßig zu verschieben oder diesen zu fixieren. Über die Einstellung des Parameters AREAnCS kann festgelegt werden, ob als Referenzkoordinatensystem das Weltkoordinatensystem genommen wird (für die gleichmäßige Verschiebung des benutzerdefinierten Bereichs) oder das Basiskoordinatensystem (für die Fixierung des benutzerdefinierten Bereichs).

Wenn der Anwender das Basiskoordinatensystems nicht ändert, verbleibt der benutzerdefinierte Bereich unverändert, egal, welches Koordinatensystem er ausgewählt hat.

Koordinatensystem	Beschreibung
Weltkoordinatensystem	Wenn das Basiskoordinatensystem verändert wird, verschiebt sich automatisch auch der benutzerdefinierte Bereich in Bezug auf das Basiskoordinatensystem. Die Änderung erfolgt in relativer Abhängigkeit der Positionen zwischen dem Roboterarm und dem benutzerdefinierten Bereich.
Basiskoordinatensystem	Die Änderung des Basiskoordinatensystem bewirkt keine gleichmäßige Verschiebung des benutzerdefinierten Bereichs in Bezug auf das Basiskoordinatensystem. Die relative Abhängigkeit der Positionen zwischen dem Roboterarm und dem benutzerdefinierten Bereich ist fixiert. Hiermit wird verhindert, dass sich die relative Beziehung zwischen den Positionen des Roboterarms und dem benutzerdefinierten Bereich ändert, sobald der Betrieb nach einer Änderung des Basiskoordinatensystems fortgesetzt wird.

Tab. 9-11: *Erläuterung des Koordinatensystems*

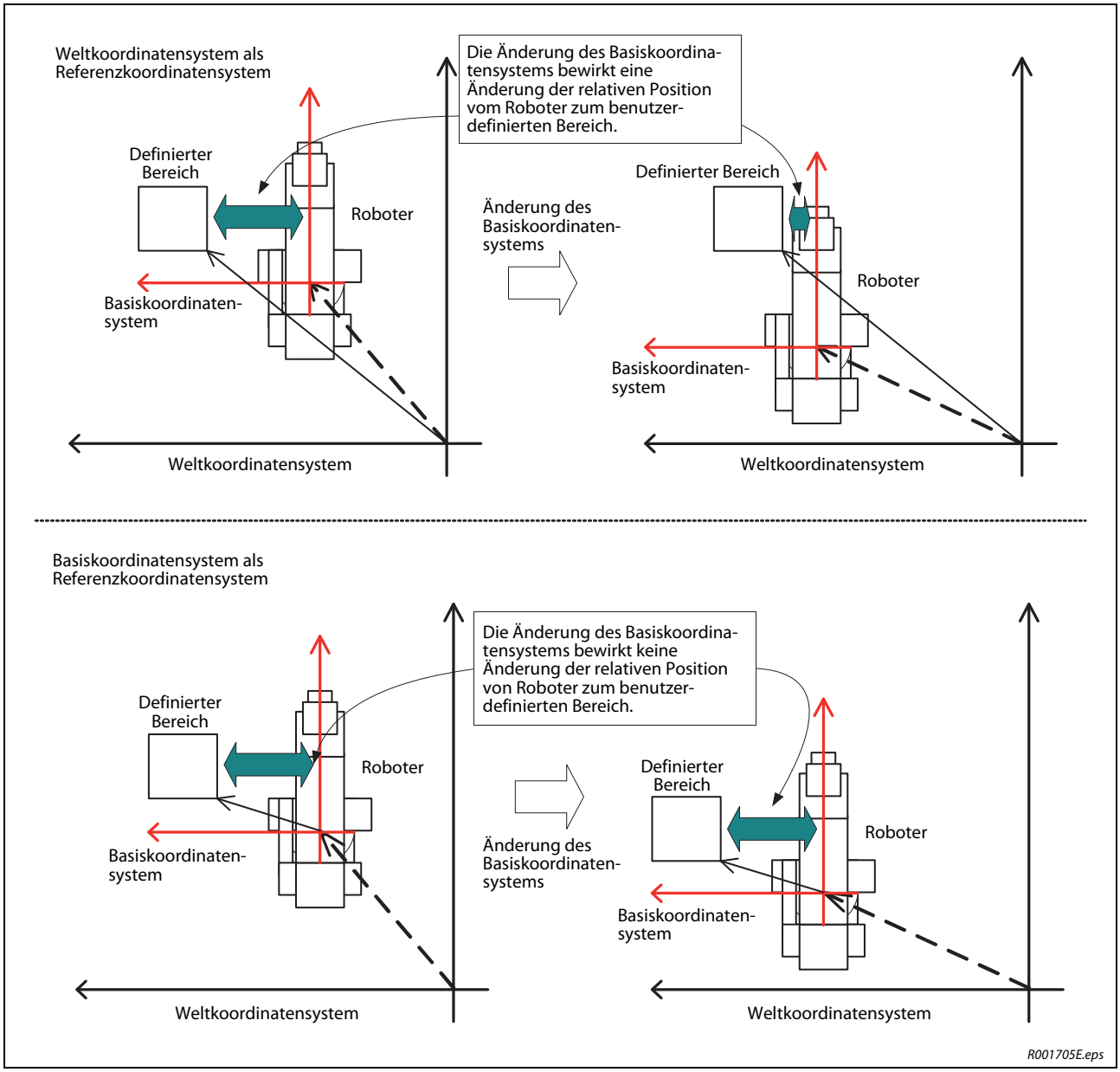


Abb. 9-7: *Unterschied zwischen Weltkoordinaten- und Basiskoordinatensystem*

9.9.2 Festlegung der Bereiche

Benutzerdefinierte Bereiche beinhalten den Positionsbereich, den Stellungsbereich und den Bereich der Zusatzachsen. Nachfolgend werden die einzelnen Schritte zur Einstellung dieser Bereiche beschrieben.

Positionsbereich

Der Positionsbereich des benutzerdefinierten Bereichs setzt sich aus den Koordinaten eines Palleteneckpunkts zusammen, die aus den Elementen X, Y und Z der Parameter AREAnP1 und AREAnP2 (n = 1 bis 32) gebildet werden.

Die festgelegten Koordinatenwerte beziehen sich auf das Koordinatensystem, das mit dem Parameter AREAnCS (n = 1 bis 32) ausgewählt wurde.

HINWEISE

Beachten Sie die folgenden Punkte, wenn Sie den Betrieb fortsetzen, nachdem Sie über den Base-Befehl o. Ä. eine Änderung am Weltkoordinatensystem vorgenommen haben und zusätzlich in den Optionen für den benutzerdefinierten Bereich ein „Basiskoordinatensystem“ ausgewählt haben:
Die in den Parametern AREAnP1 und AREAnP2 für die Elemente X, Y und Z festgelegten Koordinatenwerte müssen mit denjenigen übereinstimmen, die für das Koordinatensystem mit Parameter AREAnCS festgelegt wurden.
XYZ-Koordinatenwerte, die auf der Teaching Box, in RT ToolBox2 o. Ä. angezeigt werden, beziehen sich auf das Weltkoordinatensystem. Das bedeutet, dass die angezeigten Koordinatenwerte sich von den festgelegten Werten unterscheiden, wenn mit AREAnCS das Basiskoordinatensystem ausgewählt wird.
In diesem Fall ist es notwendig, die angezeigten Koordinatenwerte für die Einstellung in Basiskoordinaten umzuwandeln oder das Weltkoordinatensystem temporär zu initialisieren. (Das Basiskoordinatensystem und das Weltkoordinatensystem sind im Auslieferungszustand deckungsgleich.)

Die Beurteilung, ob sich das System innerhalb oder außerhalb des benutzerdefinierten Bereichs befinden, geschieht in Einheiten von 0,001 mm und 0,001 Grad. Bei einer Bewegung auf der Bereichsgrenze kann das Ergebnis unsicher werden.

Auch wenn die Elemente X, Y und Z in Parameter AREAnP1 mit denen in Parameter AREAnP2 untereinander ausgetauscht werden, bleiben die benutzerdefinierten Bereiche gleich.

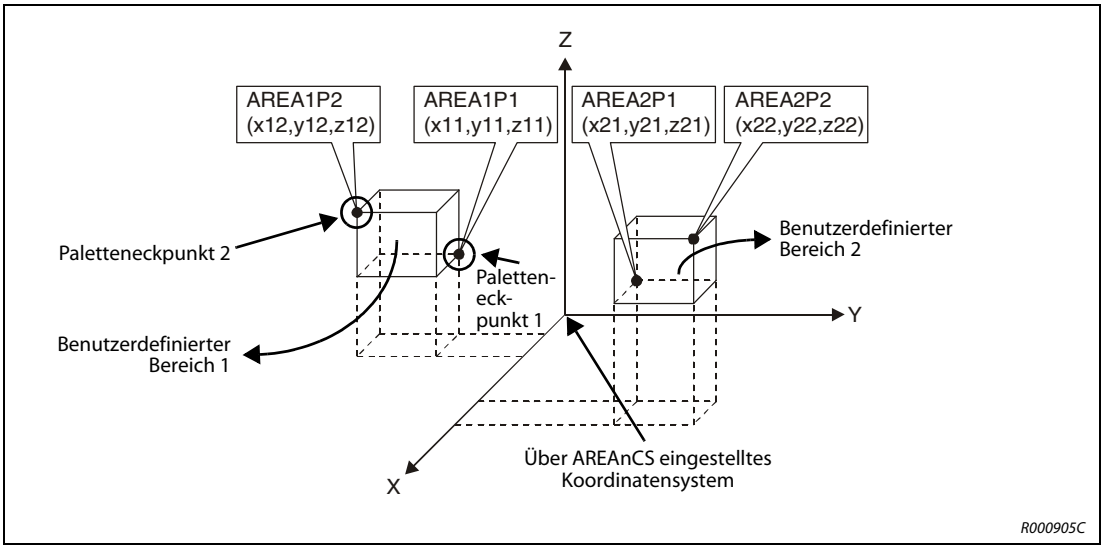


Abb. 9-8: *Positionsbereich*

Stellungsbereich

Die Elemente A, B und C in den Parametern AREAnP1 und AREAnP2 definieren im benutzerdefinierten Bereich den Stellungsbereich. Nehmen Sie die Einstellungen entsprechend dem mit AREAnCS ausgewählten Koordinatensystem vor.

ACHTUNG:

Bei einem 6-achsigen Roboter ändern sich die Koordinatenwerte von A- und C-Achse erheblich, wenn der aktuelle Koordinatenwert der B-Achse annähernd ±90 Grad ist, auch wenn nur eine leichte Stellungsbewegung auftritt. Das Vorzeichen kehrt sich um usw.
Die Ursache dafür liegt in der Steuerung des Roboters. Wenn sich die B-Achse in der Nähe von ±90 Grad befindet, kann sich der Stellungsbereich durch Beurteilung der A- und C-Achse unabhängig von Roboterbewegungen ändern, was in diesem Fall nicht anwendbar ist. Setzen Sie daher die Beurteilungsfunktion für den Stellungsbereich nur dann ein, wenn die aktuellen Koordinatenwerte der B-Achse nicht annähernd ±90 Grad sind.

HINWEISE

Beachten Sie die folgenden Punkte, wenn Sie den Betrieb fortsetzen, nachdem Sie über den Base-Befehl o. Ä. eine Änderung am Basiskoordinatensystem vorgenommen haben und zusätzlich in den Optionen für den benutzerdefinierten Bereich ein „Weltkoordinatensystem“ ausgewählt haben:
Die in den Parametern AREAnP1 und AREAnP2 für die Elemente X, Y und Z festgelegten Koordinatenwerte müssen mit denjenigen übereinstimmen, die für das Koordinatensystem mit Parameter AREAnCS festgelegt wurden.
XYZ-Koordinatenwerte, die auf der Teaching Box, in RT ToolBox2 o. Ä. angezeigt werden, beziehen sich auf das Basiskoordinatensystem. Das bedeutet, dass die angezeigten Koordinatenwerte sich von den festgelegten Werten unterscheiden, wenn mit AREAnCS das Weltkoordinatensystem ausgewählt wird.
In diesem Fall ist es notwendig, die angezeigten Koordinatenwerte für die Einstellung in Basiskoordinaten umzuwandeln oder das Basiskoordinatensystem temporär zu initialisieren. (Das Basiskoordinatensystem und das Weltkoordinatensystem werden auf den Auslieferungszustand zurückgesetzt.)

Der definierte Bereich kann sich abhängig von der relativen Lage der Elemente A, B und C unterscheiden, die in den Parametern AREAnP1 und AREAnP2 zugewiesen wurden (siehe Abb. 9-9).

Die Elemente A, B und C des Parameters AREAnP1 werden auf „–360°“ und des Parameters AREAnP2 auf „+360°“ eingestellt, wenn der Stellungsbereich nicht überprüft wird.

Die Beurteilung, ob sich das System innerhalb oder außerhalb des benutzerdefinierten Bereichs befinden, geschieht in Einheiten von 0,001 mm und 0,001 Grad. Bei einer Bewegung auf der Bereichsgrenze kann das Ergebnis unsicher werden.

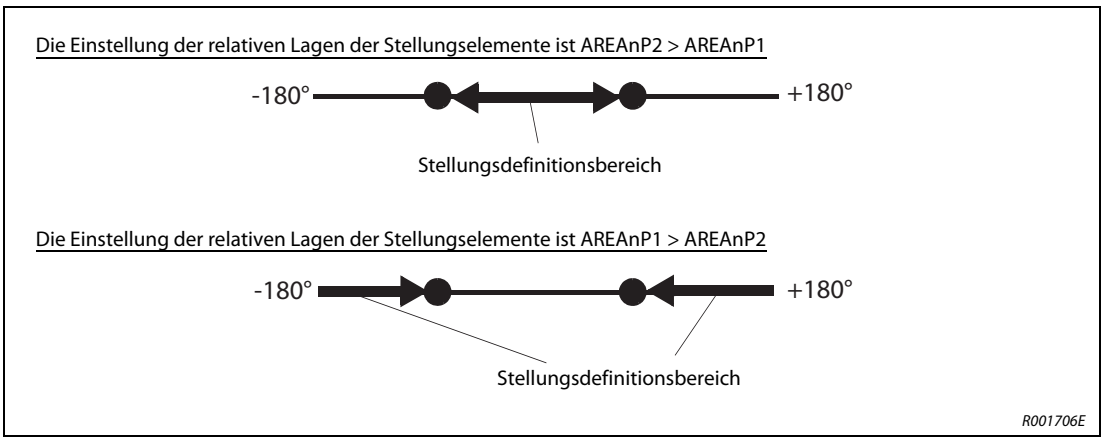


Abb. 9-9: *Relative Lage von Stellungselementen*

Zusatzachsenbereich

Der Bereich für die Zusatzachsen wird im benutzerdefinierten Bereich in den Parametern AREAnP1 und AREAnP2 mit den Elementen L1 und L2 definiert.

Sind Zusatzachsen definiert worden, werden der Positionsbereich, der Stellungsbereich und der Zusatzachsenbereich gemeinsam für den benutzerdefinierten Bereich geprüft.

HINWEISE

Die Elemente L1 und L2 in den Parametern AREAnP1 und AREAnP2 werden von dem im Parameter AREAnCS definierten Koordinatensystem nicht mit berücksichtigt.

Auch wenn die Elemente L1 und L2 in Parameter AREAnP1 mit denen in Parameter AREAnP2 untereinander ausgetauscht werden, bleiben die benutzerdefinierten Bereiche gleich.

Sind Zusatzachsen definiert, werden diese nur dann innerhalb des benutzerdefinierten Bereichs beurteilt, wenn sich Positionsbereich, Stellungsbereich und Zusatzachsenbereich zusammen im benutzerdefinierten Bereich befinden.

Die Beurteilung, ob sich das System innerhalb oder außerhalb des benutzerdefinierten Bereichs befinden, geschieht in Einheiten von 0,001 mm und 0,001 Grad. Bei einer Bewegung auf der Bereichsgrenze kann das Ergebnis unsicher werden.

Werden keine Zusatzachsen eingesetzt (Achsen J7 und J8), brauchen diese nicht definiert zu werden.

9.9.3 Mechanismusauswahl für den Begrenzungsbereich

Der Mechanismus für die Bereichsprüfung wird mit dem Parameter AREAnME ausgewählt. Standardmäßig ist der Mechanismus 1 ausgewählt (1). Stellen Sie beim Einsatz von Multimechanismen usw. die entsprechende Nummer ein.

9.9.4 Festlegung der Bereichsprüfmethode

Die Art der Bereichsprüfung, ob sich der Roboter z. B. innerhalb des benutzerdefinierten Bereichs befindet, wird mit dem Parameter AREAnAT eingestellt.

Einstellung	Benutzerdefinierter Bereich	
	Innerhalb	**Außerhalb**
0: Gesperrt	Keine Reaktion des Systems	Keine Reaktion des Systems
1: Signalausgabe und setzen der Statusvariablen	Das zugeordnete Signal USRAREA schaltet ein. Das entsprechende Bit der Systemstatusvariablen (M_Uar32, M_Uar) schaltet ein.	Das zugeordnete Signal USRAREA schaltet aus. Das entsprechende Bit der Systemstatusvariablen (M_Uar32, M_Uar) schaltet ein.
2: Fehlerausgabe	Fehler H2090 tritt auf und der Roboter stoppt. • In diesem Fall wird nur der Positionsbereich überprüft, der Stellungs- und Zusatzachsenbereich werden nicht beachtet. • Über den JOG-Betrieb kann der Roboter durch „temporäres Zurücksetzen eines Fehlers“ aus dem Bereich wieder heraus gefahren werden.	—

Tab. 9-12: *Prüfmethode für benutzerdefinierten Bereich*

Beispiel ▽

Die Parametereinstellung erfolgt so, dass bei Eindringen des Roboters in den Bereich ❶ das Signal 10 und bei Eindringen in den Bereich ❷ das Signal 11 ausgegeben wird.

Definition der Bereiche

Bereich ❶: Koordinatensystem: Weltkoordinatensystem
Stellungsbereichprüfung: nicht notwendig
Mechanismus 1: wird verwendet
Zusatzachsen: keine

Bereich ❷: Koordinatensystem: Weltkoordinatensystem
Stellungsbereichprüfung: nicht notwendig
Mechanismus 1: wird verwendet
Zusatzachsen: keine

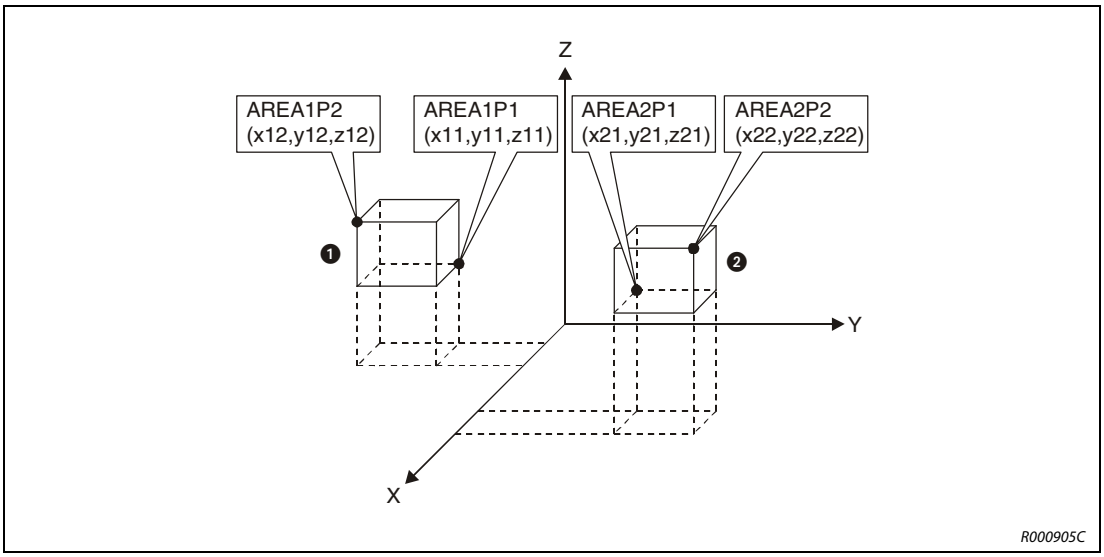


Abb. 9-10: *Beispiel für benutzerdefinierte Bereiche*

Parameter	Bedeutung	Einstellung
AREA1CS	Auswahl des Koordinatensystems für Bereich ❶	0
AREA1P1	Positionsdaten des 1. Punktes in Bereich ❶: X, Y, Z, A, B, C, L1, L2	x11, y11, z11, –360, –360, –360,0,0
AREA1P2	Positionsdaten des 2. Punktes in Bereich ❶: X, Y, Z, A, B, C, L1, L2	x12, y12, z12, 360, 360, 360,0,0
AREA1ME	Mechanismusnummer: standardmäßig 1	1
AREA1AT	Bereich ❶: Gesperrt/Signalausgabe/Fehler = 0/1/2	1
AREA2CS	Auswahl des Koordinatensystems für Bereich ❷	0
AREA2P1	Positionsdaten des 1. Punktes in Bereich ❷: X, Y, Z, A, B, C, L1, L2	x21, y21, z21, –360, –360, –360,0,0
AREA2P2	Positionsdaten des 2. Punktes in Bereich ❷: X, Y, Z, A, B, C, L1, L2	x22, y22, z22, 360, 360, 360,0,0
AREA2ME	Mechanismusnummer: standardmäßig 1	1
AREA2AT	Bereich ❷: Gesperrt/Signalausgabe/Fehler = 0/1/2	1
USRAREA	Ausgangssignal: Startnummer, Endnummer	10, 11

Tab. 9-13: *Parametereinstellungen für das Beispiel in Abb. 9-10*

△

9.10 Verfahrwegbegrenzungsebene

Die Verfahrwegsgrenzen werden über eine Ebene im Weltkoordinatensystem definiert. Bei Überschreitung dieser Bereichsgrenzen mit dem TCP des Roboters erfolgt eine Fehlermeldung.

Die Definition der Ebene erfolgt über die drei Punkte P1, P2 und P3. Dabei wird festgelegt, ob der Bereich vor – d. h. auf der Seite des Roboter-Nullpunkts – oder hinter der Ebene als Arbeitsbereich bewertet wird.

Die Funktion soll Zusammenstöße mit der Arbeitfläche oder mit umliegenden Einheiten vermeiden. Es können maximal 8 Ebenen definiert werden. Die Ebenen selbst unterliegen keiner Beschränkung.

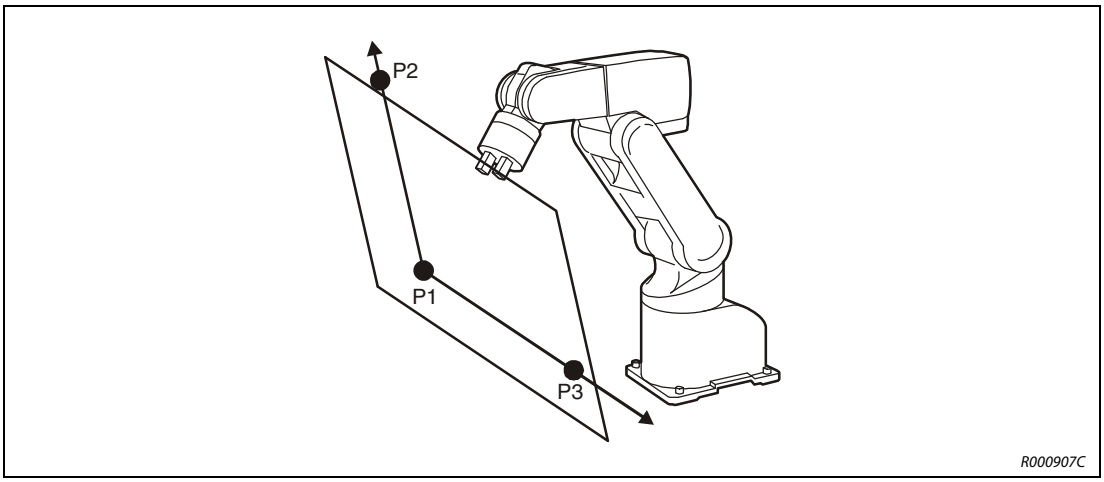


Abb. 9-11: *Verfahrwegbegrenzungsebene*

Parametereinstellung	Beschreibung
SFCnP (1 ≤ n ≤ 8)	Festlegung der 3 Punkte zur Definition der Ebene P1-Koordinaten X1, Y1 und Z1: Nullpunkt der Ebene P2-Koordinaten X2, Y2 und Z2: Position auf der X-Achse in der Ebene P3-Koordinaten X3, Y3 und Z3: Position in positiver Richtung auf der Y-Achse in der Ebene
SFCnME (1 ≤ n ≤ 8)	Standardmäßig ist der Wert auf „1“ gesetzt. Beim Betrieb mehrerer Mechanismen muss die entsprechende Mechanismusnummer gesetzt werden.
SFCnAT (1 ≤ n ≤ 8)	Freigabe der Begrenzungsebenen: 0: gesperrt 1: freigegeben (Der zugelassene Arbeitbereich liegt auf der Seite, auf der auch der Nullpunkt des Basiskoordinatensystems liegt.) –1: freigegeben (Der zugelassene Arbeitbereich liegt auf der Seite, auf der der Nullpunkt des Basiskoordinatensystems nicht liegt.

Tab. 9-14: *Parametereinstellung*

Hinweis

Die Änderung der Parametereinstellungen wird erst nach Aus- und Wiedereinschalten der Spannungsversorgung des Steuergerätes aktiv.

9.11 Automatische Rückkehr

Die Funktion bewirkt nach einer Unterbrechung des Automatik- oder des Schrittbetriebs, bei der der Roboter im JOG-Betrieb über die Teaching Box zu einer anderen Position bewegt wurde, dass die Fortsetzung des Betriebs von der Position aus erfolgt, die bei der Unterbrechung aktuell war.

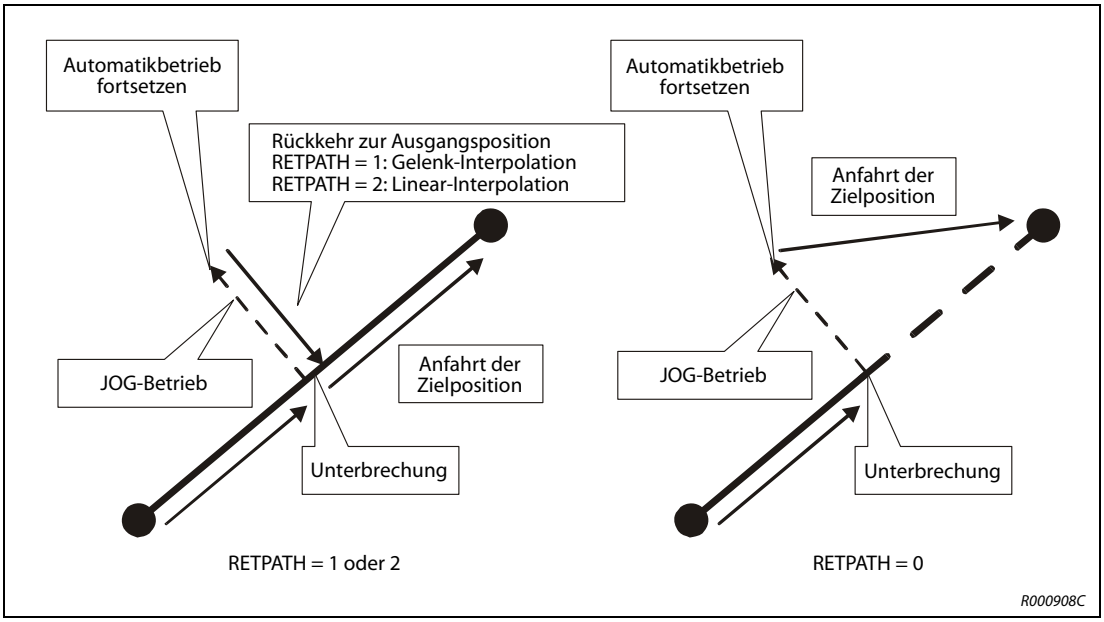


Abb. 9-12: *Automatische Rückkehr nach einer Unterbrechung*

Parametereinstellung	Beschreibung
RETPATH = 1 (Werkseinstellung)	• Rückkehr mittels Gelenk-Interpolation zu der zum Zeitpunkt der Unterbrechung aktuellen Position • Fortsetzung des Betriebs in der zum Zeitpunkt der Unterbrechung aktuellen Zeile
RETPATH = 0	Fortsetzung des Betriebs in der zum Zeitpunkt der Unterbrechung aktuellen Zeile von der im JOG-Betrieb angefahrenen Position Die Verfahrbewegung wird mit der in der Befehlzeile festgelegten Interpolationsart von der aktuellen zur Zielposition ausgeführt.
RETPATH = 2	• Rückkehr mittels Linear-Interpolation zu der zum Zeitpunkt der Unterbrechung aktuellen Position • Fortsetzung des Betriebs in der zum Zeitpunkt der Unterbrechung aktuellen Zeile

Tab. 9-15: *Parametereinstellung*

GEFAHR:

Auch wenn bei einer Einstellung des Parameters RETPATH auf „1" der JOG-Betrieb in einer anderen Interpolationsart als der Gelenk-Interpolation (XYZ-, Werkzeug-, Kreis-JOG-Betrieb usw.) ausgeführt wurde, erfolgt die Rückkehr zu der zum Zeitpunkt der Unterbrechung aktuellen Position mittels Gelenk-Interpolation. Achten Sie daher darauf, dass es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommen kann.

HINWEISE

Ist Parameter RETPATH auf „2“ eingestellt, und der Roboter wird nach einer Unterbrechung im JOG-Betrieb zu einer Position bewegt, in der die Stellungs- oder Multirotationsdaten von denen der Ursprungsposition abweichen, kann es vorkommen, dass der Roboter die Position der Unterbrechung nicht mehr anfahren kann. Fahren Sie in diesem Fall die Position der Unterbrechung an und setzen Sie den Betrieb fort.

Ist Parameter RETPATH bei aktivierter Cnt-Einstellung auf „1“ oder „2“ eingestellt, kehrt der Roboter nicht zu der Position zurück, die zum Zeitpunkt der Unterbrechung aktuell war, sondern zu einem Punkt auf der geradlinigen Verbindung zwischen den Positionen (siehe folgende Abbildung). Ist der Parameter auf „0“ eingestellt, erfolgt die Anfahrt der Zielposition direkt von der aktuellen Position aus.

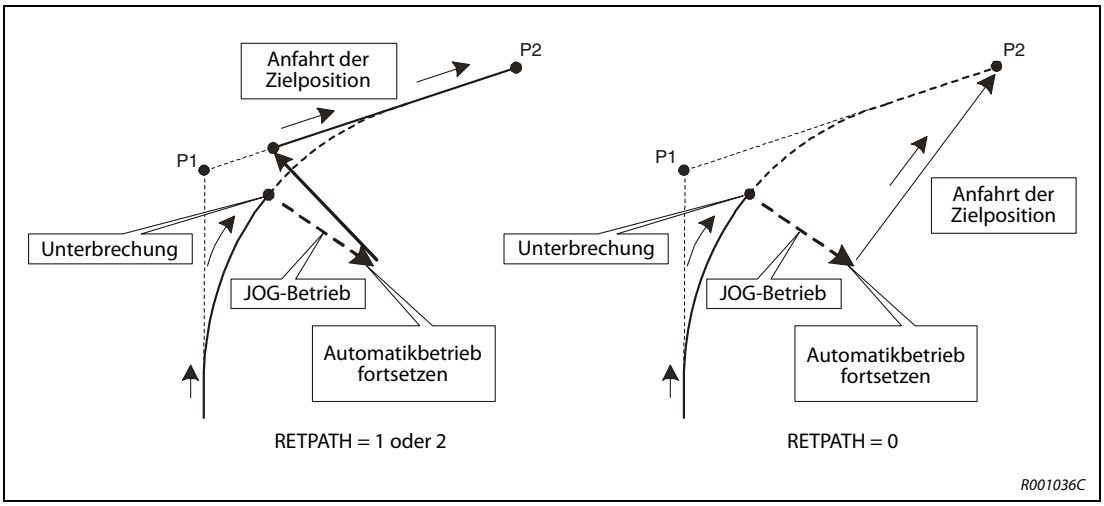


Abb. 9-13: *Automatische Rückkehr bei aktivierter Cnt-Einstellung*

9.12 Automatischer Programmstart nach dem Einschalten

Die Funktion ermöglicht den automatischen Start eines Roboterprogramms nach dem Einschalten der Spannungsversorgung.

GEFAHR:

Beachten Sie, dass der Roboterbetrieb sofort nach dem Einschalten der Spannungsversorgung startet. Verwenden Sie die Funktion daher nur nach sorgfältiger Prüfung aller betriebsrelevanten Umstände.

Parametereinstellung	Beschreibung
SLTn	Beispiel: SLT2 = 2, ALWAYS, REP Festlegung von Programmname, Startbedingung und Ausführungsformat In diesem Abschnitt wird die Startbedingung erläutert.
ALWENA	0 oder 1 Der Parameter ermöglicht in Programmen mit der Startbedingung „ALWAYS“ die Ausführung von Multitasking-Befehlen wie XRun und XLoad und des Befehls Servo.

Tab. 9-16: *Parametereinstellung*

① Erstellen Sie zuerst das ständig auszuführende Programm (Startbedingung: ALWAYS) und das Betriebsprogramm

Programm #2, Programm mit Startbedingung ALWAYS

```
1  'Automatisch startendes Beispielprogramm
2  '
3  'Führt Programm #1 aus, wenn der MODE-Umschalter auf AUTOMATIC geschaltet ist
4  'Stoppt das Programm und springt in die erste Programmzeile zurück, wenn der MODE-
   Umschalter nicht auf AUTOMATIC geschaltet ist
5  '
6  If M_Mode <> 2 AND (M_Run(1) = 1 OR MWai(1) = 1) Then GoSub *MTSTOP
7  If M_Mode = 2 AND M_Run(1) = 0 AND MWai(1) = 0 Then GoSub *MTSTART
8  If M_Mode = 1 Then Hlt 'Zum Debuggen
9  End
10 '
11 *MTSTART
12 XRun 1,"1"
13 Return
14 '
15 *MTSTOP
16 XStp 1
17 XRst 1
18 Return
```

Programm #1, Betriebsprogramm (beliebiges Programm)

```
1  'Hauptprogramm
2  Servo On
3  M_Out(8) = 0
4  Mov P1
5  M_Out(8) = 1
6  Mov P2
7  End
P1 = (+300.00,–200.00,+200.00,+0.00,+180.00,+0.00)(6,0)
P2 = (+300.00,+200.00,+200.00,+0.00,+180.00,+0.00)(6,0)
```

② Stellen Sie anschließend die Parameter entsprechend den Werten in der folgenden Tabelle ein.

Parametereinstellung	Beschreibung
SLTn	SLT2 = 2, ALWAYS, REP 'Ständige Ausführung des Programms #2
ALWENA	1 In Programmen mit der Startbedingung „ALWAYS“ ist die Ausführung von Multitasking-Befehlen wie XRun und XLoad und des Befehls Servo möglich.

Tab. 9-17: *Einstellung der Parameter*

Schalten Sie nach der Einstellung der Parameter die Spannungsversorgung des Steuergerätes aus.

③ Im oben beschriebenen Beispielprogramm wird nach Einschalten der Spannungsversorgung das Programm #1 ausgeführt und der Roboterbetrieb gestartet, wenn sich der MODE-Umschalter in der Stellung AUTOMATIC befindet.

9.13 Handgreifer

In der Werkseinstellung ist ein Handgreifer mit Doppelmagnetspule voreingestellt. Bei Einsatz einer Einfachmagnetspule oder bei Steuerung des Roboters durch die Handsensorsignale über allgemeine Eingänge muss der Parameter HANDTYPE wie im Folgenden beschrieben eingestellt werden.

In der Werkseinstellung ist der Parameter auf folgende Werte gesetzt:

Parameter	Einstellung
HANDTYPE	D900,D902,D904,D906, , , ,

Tab. 9-18: *Parametereinstellung*

Die Werte entsprechen von links beginnend: Hand #1, #2 usw. Die Werkseinstellung ist wie folgt:

Hand 1 = Zugriff auf die Eingangssignale #900 und #901
Hand 2 = Zugriff auf die Eingangssignale #902 und #903
Hand 3 = Zugriff auf die Eingangssignale #904 und #905
Hand 4 = Zugriff auf die Eingangssignale #906 und #907

Die Handnummern 1 bis 4 (oder 8) werden als Argument in den Befehlen zum Öffnen und Schließen der Hände (HOpen und HClose) verwendet.

Einstellung

Bei Einsatz eines Handgreifers mit Doppelmagnetspule muss vor der Signalnummer ein „D“ eingefügt werden. Für die Handnummern gilt: $1 \leq$ Handnummer ≤ 4.

Bei Einsatz eines Handgreifers mit Einfachmagnetspule muss vor der Signalnummer ein „S“ eingefügt werden. Für die Handnummern gilt: $1 \leq$ Handnummer ≤ 8.

Beispiel ▽

In den folgenden Beispielen wird das Magnetventil vom Anwender an die allgemeinen Signale angeschlossen.
Die Zuweisung von zwei Handgreifern mit Doppelmagnetspule an das allgemeine Eingangssignal #10 erfolgt über:
HANDTYPE = D10,D12, , , , , ,

Die Zuweisung von drei Handgreifern mit Einfachmagnetspule an das allgemeine Eingangssignal #10 erfolgt über:
HANDTYPE = S10,S11,S12, , , , ,

Die Zuweisung der Hand 1 mit Doppelmagnetspule an das allgemeine Eingangssignal #10 und der Hand 2 mit Einfachmagnetspule an das allgemeine Eingangssignal #12 erfolgt über:
HANDTYPE = D10,S12, , , , , ,

△

9.14 Handgreiferzustand nach Initialisierung

Die folgende Tabelle zeigt die Werkseinstellung des Handgreiferzustands:

Handausführung	Handgreiferzustand	Ausgangssignalzustand		
		Mechanismus #1	Mechanismus #2	Mechanismus #3
Bei installierter Schnittstellenkarte zur Steuerung der pneumatischen Greifhand (Doppelmagnetspule)	Hand 1 = geöffnet	900 = 1	910 = 1	920 = 1
		901 = 1	911 = 1	921 = 1
	Hand 2 = geöffnet	902 = 1	912 = 1	922 = 1
		903 = 1	913 = 1	923 = 1
	Hand 3 = geöffnet	904 = 1	914 = 1	924 = 1
		905 = 1	915 = 1	925 = 1
	Hand 4 = geöffnet	906 = 1	916 = 1	926 = 1
		907 = 1	917 = 1	927 = 1
Bei installierter Schnittstellenkarte zur Steuerung der motorbetriebenen Greifhand	Hand geöffnet	M_Out (9*0) bis M_Out (9*7) werden vom System verwendet und können daher vom Anwender nicht eingesetzt werden. Ein korrektes Öffnen und Schließen des Handgreifers ist über die Signale nicht möglich.		
Schnittstelle ohne installierte Schnittstellenkarte	—	M_Out (9*0) bis M_Out (9*7) sind ohne Funktion		

Tab. 9-19: *Handgreiferzustand und Handsensorsignale*

Ein Steuergerät kann mehrere Mechanismen steuern. Wird für jeden Mechanismus eine Schnittstellenkarte zur Steuerung der pneumatischen Greifhand eingesetzt, erfolgt die Zuweisung der Handsteuersignale in folgender Form:
Mechanismus #1 = #900 bis #907 (Standardanschluss mit einem Roboter an einer Steuereinheit)
Mechanismus #2 = #910 bis #917
Mechanismus #3 = #920 bis #927

Wird für den Roboter eine Schnittstellenkarte zur Steuerung einer motorbetriebenen Greifhand eingesetzt, verwendet das System die 900er Signale als festgelegte Steuersignale. Der Anwender hat keine Zugriffsmöglichkeit mehr auf diese Signale. Verwenden Sie daher die Handsteuerbefehle oder die Tasten zur Handsteuerung auf der Teaching Box.

In der Grundeinstellung sind alle Handgreifer nach dem Einschalten der Spannungsversorgung geöffnet.

Parameter	Signalnummer	Einstellung
HANDNIT	900, 901, 902, 903, 904, 905, 906, 907	1, 0, 1, 0, 1, 0, 1, 0

Tab. 9-20: *Grundeinstellung des Parameters HANDINIT*

Die Tabelle gilt für den Standardanschluss mit einem Roboter an einer Steuereinheit. Beim Anschluss mehrerer Mechanismen an eine Steuereinheit muss eine Zuordnung von Mechanismus und Ausgangssignal über den Parameter HANDINIT erfolgen.

Beispiel ▽ Soll nur die Hand 1 nach dem Einschalten der Spannungsversorgung geschlossen sein, muss der Parameter HANDINIT wie folgt eingestellt werden. Analog dazu wird eine motorbetriebene Greifhand (Handnummer = 1) mit der gezeigten Einstellung geschlossen.

Parameter	Signalnummer	Einstellung
HANDNIT	900, 901, 902, 903, 904, 905, 906, 907	0, 1, 1, 0, 1, 0, 1, 0

Tab. 9-21: *Beispieleinstellung*

△

HINWEIS

Der Parameter HANDINIT legt den Handgreiferzustand nach der Initialisierung über die 900er Signale fest. Soll die Einstellung des Handgreiferzustandes nach der Initialisierung über allgemeine E/A (andere als die 900er Signale) (im Parameter HANDTYPE sind andere Signale als die 900er festgelegt) erfolgen, verwenden Sie zur Festlegung der Zustände nicht den Parameter HANDINIT, sondern die Parameter ORSTn. Die über die Parameter ORSTn gesetzten Ausgangsbitmuster entsprechen dem Initialisierungszustand der Signale nach dem Einschalten der Spannungsversorgung.

GEFAHR:

Beachten Sie, dass ein Werkstück aus einer Hand, die nach dem Einschalten der Spannungsversorgung geöffnet wird, herunterfallen kann. Es besteht Verletzungsgefahr!

9.15 Ausgangsbitmuster

In der Werkseinstellung werden alle allgemeinen Ausgangssignale nach dem Einschalten ausgeschaltet, d. h. auf „0“ gesetzt. Die Signalzustände können über Parameter geändert werden. Beachten Sie, dass eine Änderung des Parameters auch die Ausgangssignalmuster beim Zurücksetzen über einen Eingang oder über die Clr-Anweisung beeinflusst.

	Parameter	Einstellung			
Dezentrale E/A	ORST0	Signalnummer 0---------7 00000000,	8---------15 00000000,	16--------23 00000000,	24--------31 00000000
	ORST32	32--------40 00000000,	41--------49 00000000,	50--------57 00000000,	58--------66 00000000
	ORST64	00000000,	00000000,	00000000,	00000000
	ORST96	00000000,	00000000,	00000000,	00000000
	ORST0128	00000000,	00000000,	00000000,	00000000
	ORST0160	00000000,	00000000,	00000000,	00000000
	ORST192	00000000,	00000000,	00000000,	00000000
	ORST224	00000000,	00000000,	00000000,	00000000
PROFIBUS-Schnittstellenkarte	ORST2000	00000000,	00000000,	00000000,	00000000
	ORST2032	00000000,	00000000,	00000000,	00000000
	ORST2064	00000000,	00000000,	00000000,	00000000
	ORST2096	00000000,	00000000,	00000000,	00000000
	ORST2128	00000000,	00000000,	00000000,	00000000
	ORST2160	00000000,	00000000,	00000000,	00000000
	ORST2192	00000000,	00000000,	00000000,	00000000
	ORST2224	00000000,	00000000,	00000000,	00000000
	ORST2256	00000000,	00000000,	00000000,	00000000
	ORST2288	00000000,	00000000,	00000000,	00000000
	:	:			
	:	:			
	:	:			
	ORST5008	00000000,	00000000,	00000000,	00000000
	ORST5040	00000000,	00000000,	00000000,	00000000

Tab. 9-22: *Ausgangsbitmuster (1)*

	Parameter	Einstellung			
CC-Link-Schnittstellenkarte	ORST6000	00000000,	00000000,	00000000,	00000000
	ORST6032	00000000,	00000000,	00000000,	00000000
	ORST6064	00000000,	00000000,	00000000,	00000000
	ORST6096	00000000,	00000000,	00000000,	00000000
	ORST6128	00000000,	00000000,	00000000,	00000000
	ORST6160	00000000,	00000000,	00000000,	00000000
	ORST6192	00000000,	00000000,	00000000,	00000000
	ORST6224	00000000,	00000000,	00000000,	00000000
	ORST6256	00000000,	00000000,	00000000,	00000000
	ORST6288	00000000,	00000000,	00000000,	00000000
	:	:			
	:	:			
	:	:			
	ORST7984	00000000,	00000000,	00000000,	00000000
	ORST8016	00000000,	00000000,	00000000,	00000000
Verbindung zur SPS ①	ORST10000	10000---10007 00000000,	10008---10015 00000000,	10016---10023 00000000,	10024---10031 00000000
	ORST10032	10032---10039 00000000,	10040---10047 00000000,	10048---10055 00000000,	10056---10063 00000000
	ORST10064	00000000,	00000000,	00000000,	00000000
	ORST10096	00000000,	00000000,	00000000,	00000000
	ORST10128	00000000,	00000000,	00000000,	00000000
	:	:			
	:	:			
	:	:			
	ORST18160	00000000,	00000000,	00000000,	00000000

Tab. 9-22: *Ausgangsbitmuster (2)*

① Die Verbindung zur SPS gilt nur für die Steuergeräte CRnQ

Folgende Einstellungen sind möglich:
0 = AUS
1 = EIN
* = Status beibehalten (wird beim Einschalten ausgeschaltet)

Beispiel ▽ Die allgemeinen Ausgangssignale 10138, 10139, 10140, 10160, 10161 und 10168 der Standardschnittstelle sollen nach Einschalten der Spannungsversorgung eingeschaltet werden.

Parameter	Einstellung			
ORST10128	10128---10135 00000000, 11000000,	10036---10143 00000000, 00111000,	10144---10151 00000000, 00000000,	10152---10159 00000000.....Werkseinstellung 00000000.....Einstellwert
ORST10160	10160---10167 00000000, 11000000,	10168---10175 00000000, 10000000,	10176---10183 00000000, 00000000,	10184---10191 00000000.....Werkseinstellung 00000000.....Einstellwert

Tab. 9-23: *Beispieleinstellung der Parameter*

Sollen zusätzlich die Signale 10148, 10143 und 10150 beim Zurücksetzen der allgemeinen Ausgangssignale ihren vorherigen Zustand beibehalten, sind folgende Einstellungen vorzunehmen:

Parameter	Einstellung			
ORST10128	00000000,	00111000,	0000***0,	00000000
ORST10160	11000000,	10000000,	00000000,	00000000

Tab. 9-24: *Zusätzliche Parametereinstellung*

Nach Einstellung der Parameter werden die Signale 10148, 10149, 10150 beim Einschalten der Spannungsversorgung auf „0" (AUS) gesetzt. Es ist nicht möglich, die Einstellung so vorzunehmen, dass die Signale nach dem Einschalten der Spannungsversorgung den Zustand „1" (EIN) annehmen und diesen Zustand dann nach Zurücksetzen der allgemeinen Ausgangssignale beibehalten.

△

HINWEIS

Geben Sie bei der Einstellung der Parameter die richtige Anzahl von Nullen ein, da ansonsten beim Einschalten der Spannungsversorgung eine Fehlermeldung erfolgt.

9.16 Kommunikationseinstellungen (RS232)

9.16.1 Allgemeine Beschreibung

Die Steuergeräte der Robotermodelle CRnD besitzen standardmäßig eine RS232-Schnittstelle.

Üblicherweise dient diese Schnittstelle zur Datenverbindung mit einem Personal Computer, um Roboterprogramme zu übertragen und für Test und Fehlersuche (Debugging) über die optionale Programmier-Software RT ToolBox2. Weiterhin können über die Schnittstelle Parameter eingestellt werden und sie kann zur Datenkommunikation mit externen Geräten verwendet werden. Die Kommunikationsvorgänge werden durch Kommunikationsbefehle (Open, Close, Print, Input usw.) im Roboterprogramm gesteuert. Dies wird als Daten-Link (Datenverbindung) bezeichnet.

Das Steuergerät kann nicht durch externe Geräte wie z. B. einen PC (mit Ausnahme des Automatikbetriebs und der Statusanzeige) gesteuert werden. Sollte eine Steuerung durch externe Geräte erforderlich sein, wenden Sie sich an Ihren Vertriebspartner.

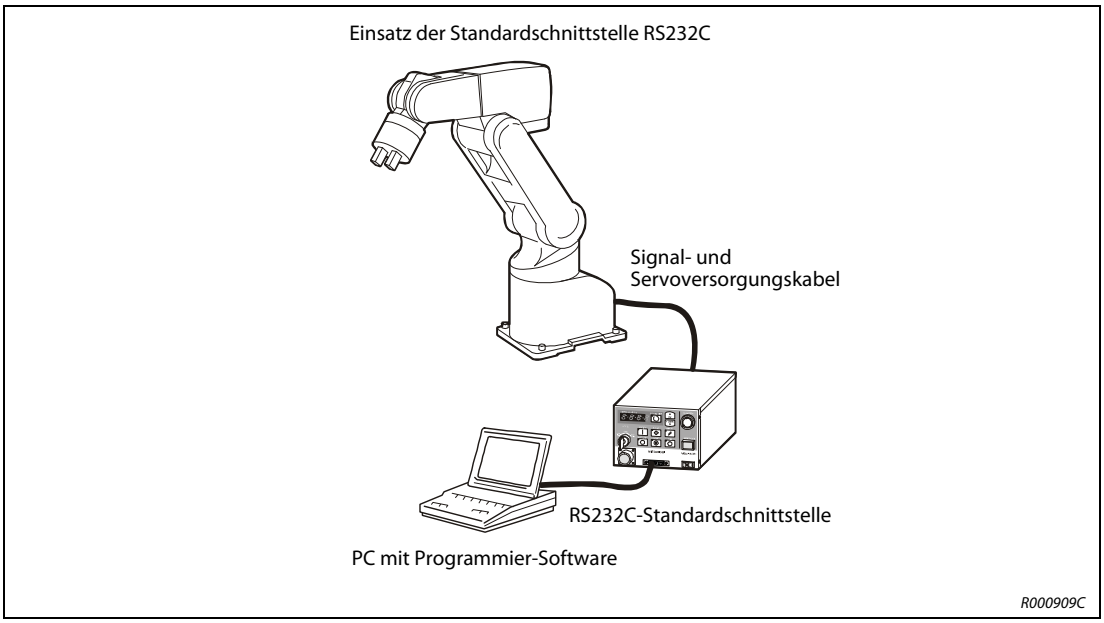


Abb. 9-14: *Beispiel für eine Schnittstellenbelegung*

9.17 Kommunikationseinstellungen (Ethernet)

9.17.1 Vorgaben der CE-Norm für die Anschlussleitung

Um die EMV-Anforderungen der CE-Norm zu erfüllen, muss an die Ethernet-Verbindungsleitung ein Entstörfilter und Ferritkern installiert werden. Das Entstörfilter und der Ferritkern sind Bestandteile des Lieferumfangs.

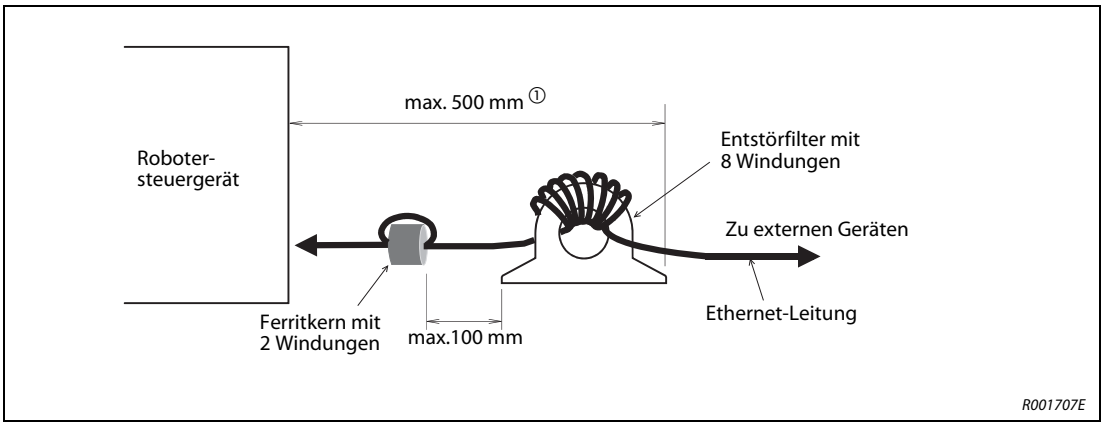


Abb. 9-15: *Installation von Entstörfilter und Ferritkern an der Ethernet-Verbindungsleitung*

① CR1D/CR2D: Maximale Länge vom Anschlusstecker des Steuergerätes aus gesehen
CR3D: Maximale Länge vom Gehäuse des Steuergerätes aus gesehen

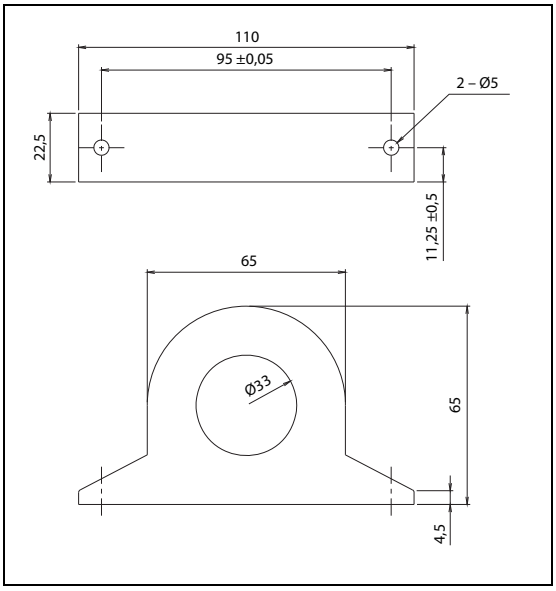


Abb. 9-16:
Abmessungen des Entstörfilters

9.17.2 Parametereinstellungen

NETIP (IP-Adresse der Robotersteuerung)

Mit dem Parameter NETIP wird die IP-Adresse der Robotersteuerung eingestellt.
Die IP-Adresse setzt sich aus vier Zahlen von jeweils 0 bis 255 zusammen, die mit einem Punkt voneinander getrennt sind.

Beispiel für eine IP-Adresse: 192.168.0.1

Wenn die Steuerung und der PC direkt miteinander verbunden sind, ohne in ein LAN-Netzwerk eingebunden zu sein, ist es möglich die zufällig vorgegebene IP-Adresse zu verwenden. Bei Anschluss an ein hausinternes LAN-Netzwerk muss die IP-Adresse verwendet werden, die vom IT-Administrator zugewiesen wird. Wenn für mehrere Geräte die gleiche IP-Adresse eingestellt wird, kann eine ordnungsgemäße Funktion nicht mehr gewährleistet werden.

Der PC, der mit dem Steuergerät Daten austauschen soll, muss an das gleiche Netzwerk, wie das Steuergerät angeschlossen sein.

NETMSK (Subnetzmaske)

Mit dem Parameter NETMSK wird die Subnetzmaske der Robotersteuerung eingestellt.
Neben der IP-Adresse dient die Einstellung der Subnetzmaske zur Festlegung eines Unternetzwerks. Die Subnetzmaske setzt sich aus vier Zahlen von jeweils 0 bis 255 zusammen, die mit einem Punkt voneinander getrennt sind.

Beispiele für eine Subnetzmaske: 255.255.255.0 oder 255.255.0.0

Bei einer Direktverbindung zwischen PC und Steuergerät ist es möglich, die vorgegebene Subnetzmaske zu verwenden. Bei Anschluss an ein hausinternes LAN-Netzwerk muss die Subnetzmaske eingestellt werden, die vom IT-Administrator zugewiesen wird.

NETPORT (Schnittstellennummer)

Mit dem Parameter NETPORT wird die Schnittstellennr. der Robotersteuerung eingestellt.
Die Schnittstellennr. besteht aus 9 Elementen, die über einen Wert dargestellt werden. Das erste Element (Elementnr. 1) wird für externe Steuerungsfunktionen in Echtzeit, die restlichen Elemente zwei bis neun (Elementnr. 2 bis 9) werden für die Programmier-Software oder für die Datenverbindung (Daten-Link) verwendet.

Im Normalfall müssen die vorgegebene Schnittstellennummer nicht geändert werden. Vermeiden Sie die doppelte Verwendung von Schnittstellennummern.

CPRCE11 bis 19 (Protokoll)

Der Parameter CPRCEn muss eingestellt werden, wenn die Datenverbindung (Data-link) verwendet werden soll.

Hiermit wird das Protokoll der Kommunikation festgelegt. Es sind drei Einstellungen des Protokolls möglich:

0:	kein Protokoll:	Das Protokoll der Software RT ToolBox2 wird verwendet
1:	Protokoll:	Nicht verwendet (Vermeiden Sie eine versehentliche Einstellung auf diesen Wert!)
2:	Data-link-Verbindung:	Dieses Protokoll erlaubt die Verwendung der Kommunikationsbefehle Open/Input/Print

COMDEV (Gerätezuordnung für COM1: bis 8)

Der Parameter COMDEV muss eingestellt werden, wenn die Datenverbindung (Daten-Link) verwendet werden soll und legt fest, welches Gerät über welche Kommunikationsleitung COM1: bis 8 angesprochen wird. COM1: bis 8 wird in Zusammenhang mit dem Open-Befehl des Roboterprogramms verwendet.

Der Parameter COMDEV darf nur dann verwendet werden, wenn der Parameter CPRCEn auf „Datenverbindung" (Data-link) eingestellt ist.

Die Einstellwerte der Ethernet-Schnittstellenoption entsprechen den Schnittstellennummern, die mit dem Parameter NETPORT eingestellt werden.

In der folgenden Tabelle kennzeichnet n die Elementnummer des Parameters COMDEV(n)

n	Mit COMDEV(n) eingestellter Gerätename	Schnittstellennummer
1	OPT11	Mit NETPORT (2) festgelegte Schnittstellennummer
2	OPT12	Mit NETPORT (3) festegelegte Schnittstellennummer
3	OPT13	Mit NETPORT (4) festegelegte Schnittstellennummer
4	OPT14	Mit NETPORT (5) festegelegte Schnittstellennummer
5	OPT15	Mit NETPORT (6) festegelegte Schnittstellennummer
6	OPT16	Mit NETPORT (7) festegelegte Schnittstellennummer
7	OPT17	Mit NETPORT (8) festegelegte Schnittstellennummer
8	OPT18	Mit NETPORT (9) festegelegte Schnittstellennummer
9	OPT19	Mit NETPORT (10) festegelegte Schnittstellennummer

Tab. 9-25: *Zuordnung der Einstellung von COMDEV zur Schnittstellennummer*

Im folgenden Beispiel wird die mit NETPORT(4) festgelegte Schnittstellennnummern der Datenverbindung über COM3: zugeordnet.
COMDEV(3) = OPT13 → Einstellung von OPT13 als drittes Element von COMDEV
CPRCE13 = 2 → Protokolleinstellung auf „Datenverbindung" (Daten-Link)

NETMODE

Der Parameter NETMODE muss eingestellt werden, wenn die Datenverbindung (Daten-Link) verwendet werden soll.

Mit dieser Einstellung wird das Robotersteuergerät für die Datenübertragung mit TCP/IP-Kommunikation als Server oder Client festgelegt. Abhängig von der Anwendung und den an die Robotersteuerung angeschlossenen Geräten muss diese Einstellung entsprechend angepasst werden.

Die Auswahl, ob die Robotersteuerung als Server oder Client arbeitet, ist bei der Software-Version ab H7 möglich. Bei den Vorgängerversionen kann die Robotersteuerung ausschließlich als Server betrieben werden.

NETHSTIP (IP-Adresse der Zielservers für die Datenkommunikation)

Die Einstellung des Parameters NETHSTIP erfolgt, wenn die Robotersteuerung für die Datenverbindung als Client eingestellt ist. Stellen Sie die IP-Adresse des Servers ein, mit dem die Datenverbindung aufgebaut werden soll.

Die Einstellung ist nur dann nötig, wenn die Robotersteuerung mit dem Parameter NETMODE als „Client" festgelegt wird.

MXTTOUT (Zeitüberschreitungswert bei Ausführung externer Steuerungsbefehle in Echtzeit)

Der Parameter MXTTOUT kann verändert werden, wenn für die Kommunikation mit der Robotersteuerung externe Steuerungsbefehle in Echtzeit eingesetzt werden. Die Einstellung ist als Vielfaches der Zykluszeit von ca. 7,11 ms möglich.

Bei der Ausführung eines externen Steuerungsbefehls in Echtzeit wird die Zeit, in der die Robotersteuerung keine Daten vom PC empfängt, als Timeout aufsummiert. Erreicht die Timeout-Zeit den mit MXTTOUT eingestellten Wert, stoppt der Betrieb und ein Fehler (#7820) wird ausgegeben. Soll beispielsweise nach ca. 7 Sekunden ein Fehler ausgegeben werden, muss für MXTTOUT der Wert 1000 eingestellt werden.

Der Parameter ist standardmäßig auf –1 eingestellt, wodurch die Timeout-Funktion deaktiviert ist.

9.17.3 Beispieleinstellung für Parameter 1

Das nachfolgende Beispiel zeigt, wie die folgenden Einstellungen mit der Software RT ToolBox2 vorgenommen werden:

Parameter	Einstellung
IP-Adresse der Robotersteuerung	192.168.0.20 (Parameter NETIP muss auch auf diesen Wert angepasst werden!)
IP-Adresse des Personal Computers	192.168.0.2
Schnittstellennummer der Robotersteuerung	10001

Tab. 9-26: *Einstellungen für das Beispiel 1*

Stellen Sie die Parameter der Robotersteuerung wie folgt ein. Bei Verwendung der Standardwerte braucht nichts geändert zu werden.

Parameter	Vor/nach der Änderung	Einstellung
NETIP	Vor	192.168.0.20
	Nach	192.168.0.20 (Standardwert)
NETPORT	Vor	10001
	Nach	10001 (Standardwert)

Tab. 9-27: *Parametereinstellungen der Robotersteuerung*

Stellen Sie die IP-Adresse des Personal Computers auf 192.168.0.2. Die Einstellung erfolgt abhängig vom verwendeten Betriebssystem über das Einstellmenü für die Netzwerkeigenschaften.

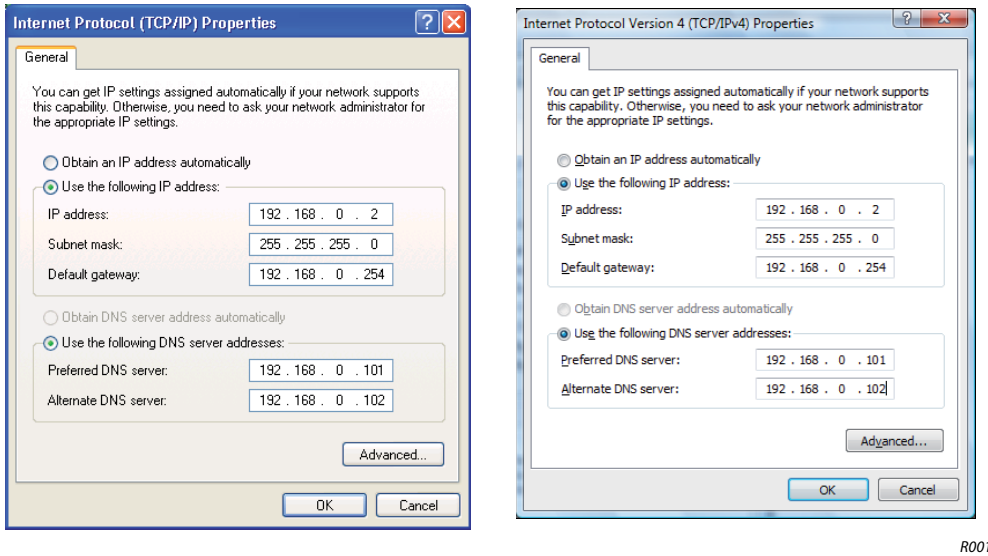

Internet Protocol (TCP/IP) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
Internet Protocol Version 4 (TCP/IPv4) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
R001709E

Abb. 9-17: *Einstellung der Netzwerkeigenschaften mit Windows XP (links) und Windows Vista (rechts)*

Die Einstellung der IP-Adresse des PCs erfolgt über das Menü „Eigenschaften von Internetprotokoll TCP/IP“ der Netzwerkverbindungen. Das Menü kann je nach Windows-Version vom o. a. Menü abweichen. Beachten Sie dazu die Bedienungsanleitung Ihrer Windows-Version.

Beachten Sie ebenfalls die Bedienungsanleitung der Software RT ToolBox2, wie diese auf Ihrem PC-System konfiguriert wird und welche Funktionen damit ausgeführt werden können.

9.17.4 Beispieleinstellung für Parameter 2-1

In diesem Beispiel hat die Robotersteuerung für die Datenverbindung die Funktion des Servers:

Parameter	Einstellung
IP-Adresse der Robotersteuerung	192.168.0.20
IP-Adresse des Personal Computers	192.168.0.2
Schnittstellennummer der Robotersteuerung	10003
Kommunikationsleitungsnummer COM-Nr. für Open-Befehl	COM3:

Tab. 9-28: *Einstellungen für das Beispiel 2-1*

Parameter	Vor/nach der Änderung	Einstellung
NETIP	Vor	192.168.0.20
	Nach	192.168.0.20 (Standardwert)
NETPORT	Vor	10000,10001,10002,10003,10004,10005,10006,10007,10008,10009
	Nach	" (Standardwert)
CPRCE13	Vor	0
	Nach	2
COMDEV	Vor	RS232, , , , , ,
	Nach	RS232, , OPT13, , , ,

Tab. 9-29: *Änderungen der Einstellungen für das Beispiel 2-1*

Stellen Sie die IP-Adresse des Personal Computers auf 192.168.0.2. Die Einstellung erfolgt abhängig vom verwendeten Betriebssystem über das Einstellmenü für die Netzwerkeigenschaften.

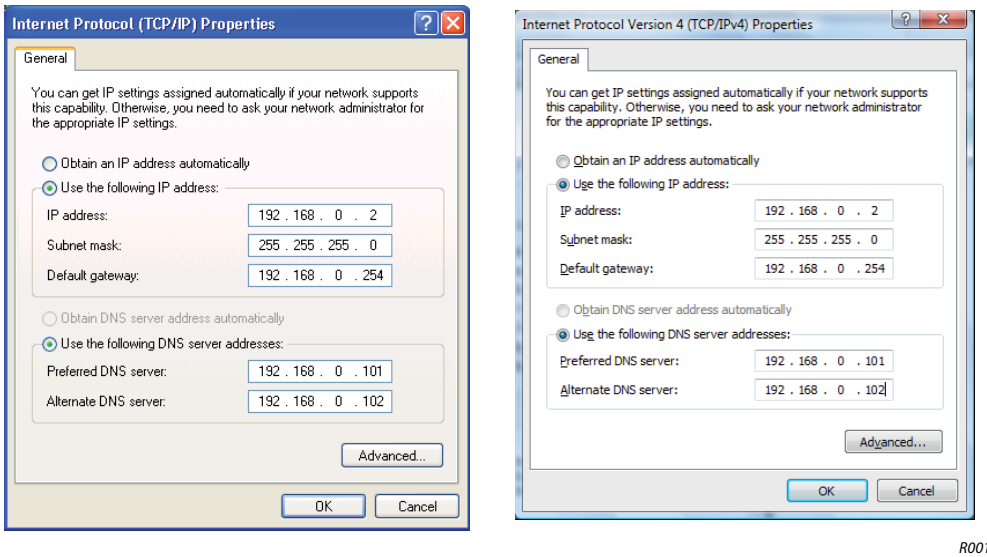

Internet Protocol (TCP/IP) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
Internet Protocol Version 4 (TCP/IPv4) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
R001709E

Abb. 9-18: *Einstellung der Netzwerkeigenschaften mit Windows XP (links) und Windows Vista (rechts)*

Die Einstellung der IP-Adresse des PCs erfolgt über das Menü „Eigenschaften von Internetprotokoll TCP/IP“ der Netzwerkverbindungen. Das Menü kann je nach Windows-Version vom o. a. Menü abweichen. Beachten Sie dazu die Bedienungsanleitung Ihrer Windows-Version.

Beachten Sie ebenfalls die Bedienungsanleitung der Software RT ToolBox2, wie diese auf Ihrem PC-System konfiguriert wird und welche Funktionen damit ausgeführt werden können.

9.17.5 Beispieleinstellung für Parameter 2-2

In diesem Beispiel hat die Robotersteuerung für die Datenverbindung die Funktion des Clients:

Parameter	Einstellung
IP-Adresse der Robotersteuerung	192.168.0.20
IP-Adresse des Personal Computers	192.168.0.2
Schnittstellennummer der Robotersteuerung	10003
Kommunikationsleitungsnummer COM-Nr. für Open-Befehl	COM3:

Tab. 9-30: *Einstellungen für das Beispiel 2-2*

Parameter	Vor/nach der Änderung	Einstellung
NETIP	Vor	192.168.0.20
	Nach	192.168.0.20 (Standardwert)
NETPORT	Vor	10000,10001,10002,10003,10004,10005,10006,10007,10008,10009
	Nach	" (Standardwert)
CPRCE13	Vor	0
	Nach	2
COMDEV	Vor	RS232, , , , , ,
	Nach	RS232, , OPT13, , , , ,
NETMODE	Vor	1,1,1,1,1,1,1,1,1,1
	Nach	1,1,0,1,1,1,1,1,1,1
NETHSTIP	Vor	192.168.0.2, 192.168.0.3, 192.168.0.4, 192.168.0.5, 192.168.0.6, 192.168.0.7, 192.168.0.8, 192.168.0.9, 192.168.0.10
	Nach	192.168.0.2, 192.168.0.3, 192.168.0.2, 192.168.0.5, 192.168.0.6, 192.168.0.7, 192.168.0.8, 192.168.0.9, 192.168.0.10

Tab. 9-31: *Änderungen der Einstellungen für das Beispiel 2-2*

Stellen Sie die IP-Adresse des Personal Computers auf 192.168.0.2. Die Einstellung erfolgt abhängig vom verwendeten Betriebssystem über das Einstellmenü für die Netzwerkeigenschaften.

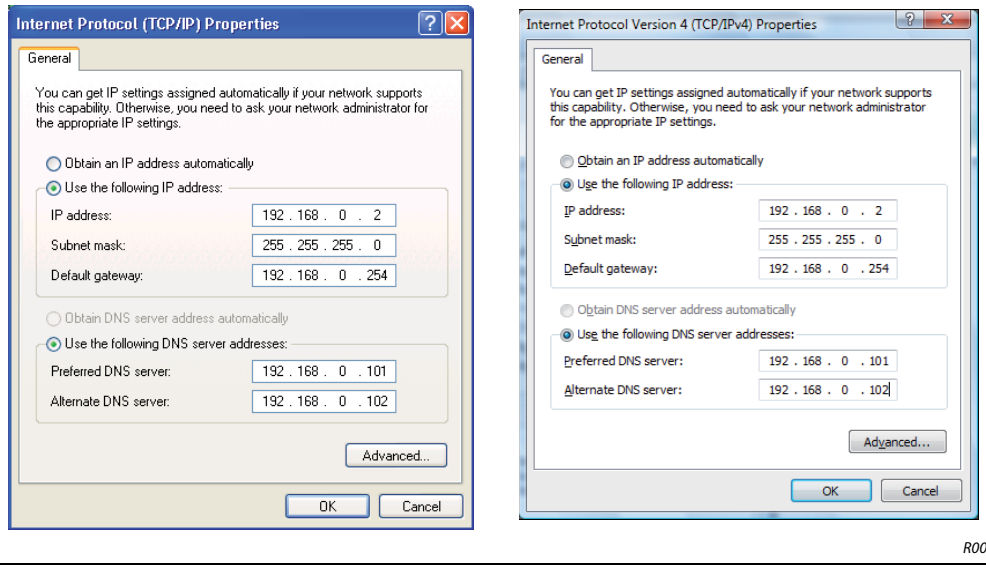

Internet Protocol (TCP/IP) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
Internet Protocol Version 4 (TCP/IPv4) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
R001709E

Abb. 9-19: *Einstellung der Netzwerkeigenschaften mit Windows XP (links) und Windows Vista (rechts)*

Die Einstellung der IP-Adresse des PCs erfolgt über das Menü „Eigenschaften von Internetprotokoll TCP/IP“ der Netzwerkverbindungen. Der Bildschirm kann je nach Windows-Version vom o. a. Menü abweichen. Beachten Sie dazu die Bedienungsanleitung Ihrer Windows-Version.

Beachten Sie ebenfalls die Bedienungsanleitung der Software RT ToolBox2, wie diese auf Ihrem PC-System konfiguriert wird und welche Funktionen damit ausgeführt werden können.

9.17.6 Beispieleinstellung für Parameter 3

Das Beispiel zeigt die Einstellungen für die Verwendung der externen Steuerungsfunktion in Echtzeit:

Parameter	Einstellung
IP-Adresse der Robotersteuerung	192.168.0.20
IP-Adresse des Personal Computers	192.168.0.2
Schnittstellennummer der Robotersteuerung	10000

Tab. 9-32: *Einstellungen für das Beispiel 3*

Parameter	Vor/nach der Änderung	Einstellung
NETIP	Vor	192.168.0.20
	Nach	192.168.0.20 (Standardwert)
NETPORT	Vor	10000,10001,10002,10003,10004,10005,10006,10007,10008,10009
	Nach	" (Standardwert)
MXTTOUT	Vor	–1
	Nach	" (Standardwert)
MXTCOM1	Vor	192.168.0.2
	Nach	" (Standardwert)

Tab. 9-33: *Änderungen der Einstellungen für das Beispiel 3*

Stellen Sie die IP-Adresse des Personal Computers auf 192.168.0.2. Die Einstellung erfolgt abhängig vom verwendeten Betriebssystem über das Einstellmenü für die Netzwerkeigenschaften.

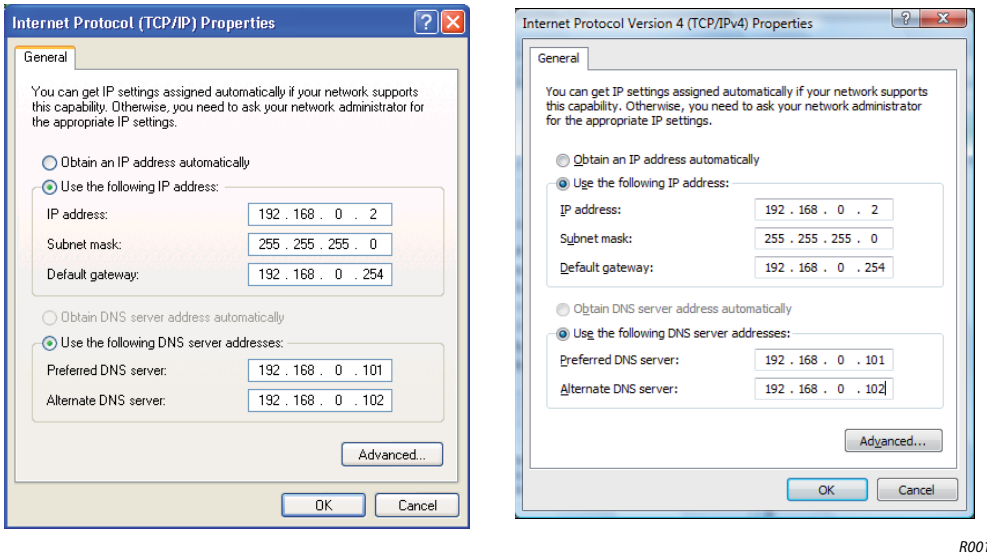

Internet Protocol (TCP/IP) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
Internet Protocol Version 4 (TCP/IPv4) Properties
General
You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.
Obtain an IP address automatically
Use the following IP address:
IP address: 192 . 168 . 0 . 2
Subnet mask: 255 . 255 . 255 . 0
Default gateway: 192 . 168 . 0 . 254
Obtain DNS server address automatically
Use the following DNS server addresses:
Preferred DNS server: 192 . 168 . 0 . 101
Alternate DNS server: 192 . 168 . 0 . 102
Advanced...
OK
Cancel
R001709E

Abb. 9-20: *Einstellung der Netzwerkeigenschaften mit Windows XP (links) und Windows Vista (rechts)*

Die Einstellung der IP-Adresse des PCs erfolgt über das Menü „Eigenschaften von Internetprotokoll TCP/IP“ der Netzwerkverbindungen. Das Menü kann je nach Windows-Version vom o. a. Menü abweichen. Beachten Sie dazu die Bedienungsanleitung Ihrer Windows-Version.

Beachten Sie ebenfalls die Bedienungsanleitung der Software RT ToolBox2, wie diese auf Ihrem PC-System konfiguriert wird und welche Funktionen damit ausgeführt werden können.

9.18 Überprüfen der Verbindung

Überprüfen Sie zuerst die folgenden Punkte, bevor Sie die Verbindung in Betrieb nehmen.

Nr.	Überprüfungspunkt	Geprüft?
1	Ist die Teaching Box sicher befestigt?	
2	Ist das Ethernet-Kabel zwischen Robotersteuerung und PC korrekt angeschlossen und sicher befestigt?	
3	Wurde das richtige Ethernet-Kabel verwendet? (Ein gekreuztes Kabel (Cross-Kabel) muss verwendet werden, wenn die Robotersteuerung und der PC direkt miteinander verbunden werden. Bei der Ethernet-Verbindung über einen HUB muss ein ungekreuztes Kabel (Patch-Kabel) eingesetzt werden.)	
4	Sind die Parameter der Robotersteuerung richtig eingestellt? (Siehe auch Abschn. 2.4)	
5	Wurde die Spannungsversorgung der Robotersteuerung mindestens einmal aus- und wieder eingeschaltet, nachdem die Parameter eingestellt wurden?	

Tab. 9-34: *Prüfpunkte zur Inbetriebnahme der Kommunikation*

9.18.1 Prüfung der Verbindung mit dem ping-Befehl von Windows

Nachfolgend wird gezeigt, wie die Ethernet-Verbindung mit dem ping-Befehl von Windows überprüft werden kann.

Wählen Sie in Windows über das Menü ***Start*** → ***Programme*** → ***MS-DOS Prompt*** den Bildschirm für die „MS-DOS-Eingabeaufforderung“ aus.
(Menü bei Windows Vista: ***Start*** → ***Programme*** → ***Accessories*** → ***Command Prompt***)

Geben Sie den ping-Befehl gefolgt von der IP-Adresse der Robotersteuerung so ein, wie in der folgenden Abb. 9-21 gezeigt.

– Erfolgt die Kommunikation ordnungsgemäß, erscheint nach der Eingabe des ping-Befehls die Ausgabe „Reply from…“ auf dem Bildschirm.

– Ist die Kommunikation gestört, erscheint die Meldung „Request time out“.

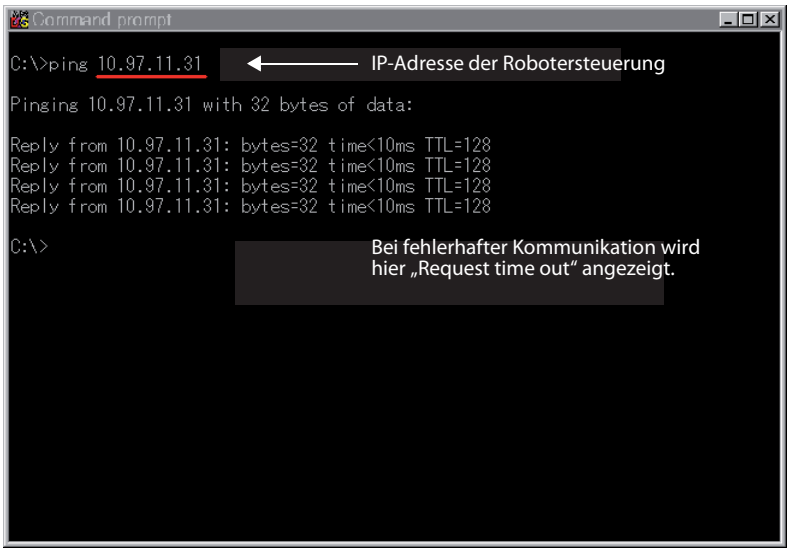

Command prompt
C:\>ping 10.97.11.31
IP-Adresse der Robotersteuerung
Pinging 10.97.11.31 with 32 bytes of data:
Reply from 10.97.11.31: bytes=32 time<10ms TTL=128
Reply from 10.97.11.31: bytes=32 time<10ms TTL=128
Reply from 10.97.11.31: bytes=32 time<10ms TTL=128
Reply from 10.97.11.31: bytes=32 time<10ms TTL=128
C:\>
Bei fehlerhafter Kommunikation wird hier „Request time out“ angezeigt.
R001710E

Abb. 9-21: *MS-DOS-Eingabebildschirm bei Ausführen des ping-Befehls*

9.19 Hand- und Werkstückbedingung

9.19.1 Optimale Beschleunigung/Abbremsung

Die Funktion erlaubt die Einstellung der optimalen Beschleunigungs-/Abbremszeit in Abhängigkeit der Last an der Handspitze. Folgende Parameter müssen zur Nutzung der optimalen Beschleunigung/Abbremsung eingestellt werden. Bei den Robotermodellen RV-SD/RH-SDH sind die Parameter auch bei Verwendung der Kollisionsüberwachung einzustellen. Dabei sind die Parameter HNDDAT0 und WRKDAT0 bei aktivierter Kollisionsüberwachung im JOG-Betrieb wirksam.

	Parameter	**Einstellung**
Hand-bedingungen	HNDDAT0	Der Parameter ist vom jeweiligen Roboter abhängig.
	HNDDAT1	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT2	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT3	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT4	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT5	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT6	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT7	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT8	Maximale Last, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
Werkstück-bedingungen	WRKDAT0	Der Parameter ist vom jeweiligen Roboter abhängig.
	WRKDAT1	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT2	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT3	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT4	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT5	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT6	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT7	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT8	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

Tab. 9-35: *Werkseinstellung der Parameter*

Die Parameter haben folgende Struktur:
Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Y, und Z

Eine Einstellung von bis zu 8 Hand- und Werkstückbedingungen ist möglich. Verwenden Sie zur Bestimmung der Handmaße einen gedachten Quader, der die Hand völlig umschließt. Die Werte für die optimale Beschleunigung/Abbremsung basieren auf den Hand- und Werkstückbedingungen, die über den LoadSet-Befehl ausgewählt werden.

9.19.2 Handgreiferzustand

Über einen weiteren Parameter kann der Handgreiferzustand bei Ausführung der Handsteuerbefehle HOpen (HClose) festgelegt werden. In Abhängigkeit des Handgreiferstatus wird die optimale Beschleunigung für den Handgreifer ohne Werkstück oder für den Handgreifer mit Werkstück berechnet.

Parameter	Einstellung
HNDHOLD1	0, 1
HNDHOLD2	0, 1
HNDHOLD3	0, 1
HNDHOLD4	0, 1
HNDHOLD5	0, 1
HNDHOLD6	0, 1
HNDHOLD7	0, 1
HNDHOLD8	0, 1

Tab. 9-36: *Werkseinstellung der Parameter*

0 = nicht schließen
1 = schließen

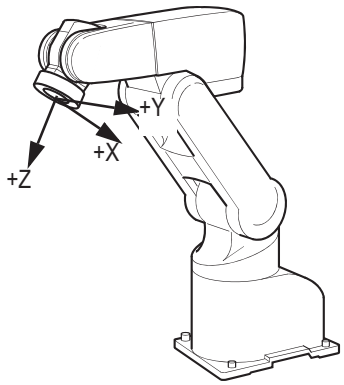
9.19.3 Definition der Koordinatensysteme für die Hand- und Werkstückbedingungen

Folgende Abbildungen zeigen die Koordinatensysteme zur Einstellung der Hand- und Werkstückbedingungen für die verschiedenen Robotermodelle. Die Koordinatensysteme für die Hand- und für die Werkstückbedingung sind identisch. Alle Größenangaben sind positiv.

Vertikaler Knickarmroboter

Das Koordinatensystem entspricht dem Werkzeugkoordinatensystem.

Folgende Elemente müssen eingestellt werden:
Schwerpunkt in X-, Y- und Z-Richtung
Größe in X-, Y- und Z-Richtung



6-achsiger Roboter

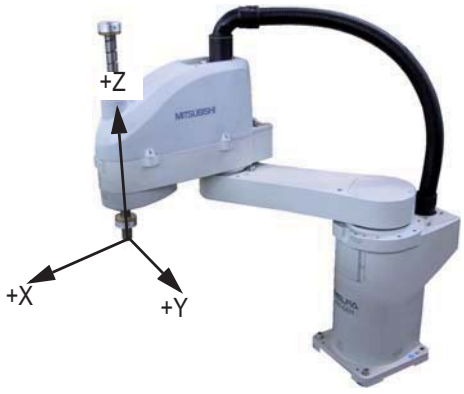
R001711E

Abb. 9-22: *Koordinatensystem für die Hand- und Werkstückbedingungen für vertikale Knickarmroboter*

SCARA-Roboter

Der Nullpunkt des Koordinatensystems liegt im unteren Ende der Kugelumlaufspindel.
Z-Achse: Die Aufwärtsrichtung wird positiv gezählt.
X-Achse: Die Bewegungsrichtung bei Armstreckung wird positiv gezählt.
Y-Achse: Die Y-Achse entspricht dem Rechtssystem.

Folgende Elemente müssen eingestellt werden:
Schwerpunkt in X-Richtung
Größe in X- und Y-Richtung



R001712E

Abb. 9-23: *Koordinatensystem für die Hand- und Werkstückbedingungen für SCARA-Roboter*

9.20 Fehlermeldung bei Erreichen des singulären Punktes

Wird der Roboter über die Teaching Box verfahren, erfolgt bei Erreichen eines singulären Punktes eine Fehlermeldung. Der Roboter kann trotz Fehlermeldung noch weiter verfahren werden, bis er einen Bereich erreicht, in dem kein Betrieb mehr möglich ist. Es erfolgt automatisch ein Rücksetzen der Fehlermeldung, wenn sich der Roboter vom singulären Punkt entfernt.

Aktivierung der Fehlermeldung

Die Fehlermeldung bei Erreichen des singulären Punktes erfolgt bei Ausführung folgender Funktionen über die Teaching Box:

- JOG-Betrieb (außer im Gelenk-JOG-Modus)
- Schrittbetrieb in Vorwärts- und Rückwärtsrichtung
- Verfahrbewegungen mit Linear-Interpolation
- direkte Befehlsausführung

Erreicht der Roboter einen singulären Punkt, ertönt ein durchgehender Signalton vom Summer im Steuergerät. Die Anzeige „STATUS.NUMBER“ des Steuergeräts ändert sich nicht. Im Falle des JOG-Betriebs erscheint parallel zum Signalton eine Warnung auf der Anzeige der Teaching Box.

Bei folgenden Funktionen erfolgt auch bei Erreichen eines singulären Punktes keine Fehlermeldung:

- JOG-Betrieb einer Zusatzachse im Gelenk-JOG-Modus über die Teaching Box
- Ausführung eines Befehls für Gelenk-Interpolation über die Teaching Box (z. B. Mov-Befehl)
- Automatikbetrieb eines Programmes
- JOG-Betrieb über spezielle Eingänge (z. B. JOGENA oder JOGM)
- Bewegung des Roboters mit gelösten Bremsen durch äußere Krafteinwirkung
- unbewegter Roboter
- TYPE-Befehl

9.21 ROM- und Highspeed-RAM-Modus

Im ROM- und Highspeed-RAM-Modus sind einige Einschränkungen beim Programmbetrieb und der Datensicherung zu beachten.

9.21.1 Übersicht

In der Werkseinstellung werden die Roboterprogramme in einem RAM (SRAM) mit Batteriepufferung gespeichert.

Dabei können jedoch Datenverluste durch eine leere Backup-Batterie und Programmfehler durch Spannungseinbrüche (z. B. kurzzeitiger Netzausfall) während eines Datenzugriffs auftreten bzw. Programme und Positionsdaten ungewollt geändert oder gelöscht werden.

Mit Hilfe von Parametern kann der Speicherort für den Datenzugriff zwischen ROM und RAM umgeschaltet werden.

Parameter	Einstellung
ROMDRV	Der Zugriff auf Programme kann zwischen RAM und ROM umgeschaltet werden. 0: RAM-Modus (Werkseinstellung) 1: ROM-Modus 2: Highspeed-RAM-Modus (Es besteht die Möglichkeit zur Auswahl des Highspeed-RAM-Modus. Dabei erfolgt der Datenzugriff über einen DRAM-Speicher.)
BACKUP	Kopiert Programme, Parameter, globale Variablen und Fehler-Logfiles vom RAM in das ROM. SRAM → FLROM (Dieser Parameter darf nicht geändert werden.) Bei einer Unterbrechung des Kopiervorgangs erfolgt die Anzeige „CANCEL“.
RESTORE	Schreibt Programme, Parameter, globale Variablen und Fehler-Logfiles vom ROM in das RAM zurück. FLROM → SRAM (Dieser Parameter darf nicht geändert werden.) Bei einer Unterbrechung des Kopiervorgangs erfolgt die Anzeige „CANCEL“.

Tab. 9-37: *Parameter zur Speicherauswahl*

Der Speicherzugriff und die somit ausführbaren Funktionen hängen von der Einstellung des Parameters ROMDRV ab. Folgende Tabelle zeigt den Zusammenhang.

Speicher	Beschreibung	Parameter ROMDRV		
		0 (RAM-Modus)	1 (ROM-Modus)	2 (Highspeed-RAM-Modus)
DRAM	Highspeed-Modus möglich Programme können ausgeführt werden, gehen aber beim Ausschalten der Versorgungsspannung verloren.		Ausführung von Programmen (Programmergebnisse gehen verloren)	Ausführung von Programmen (Programmergebnisse gehen verloren)
SRAM	Beim Ausschalten der Versorgungsspannung bleiben die Programme erhalten. Die Programme gehen erst bei entladener Batterie verloren. Es ist ein Schreib- und Lesezugriff möglich.	Ausführung von Programmen (Programmergebnisse werden gespeichert) Programmverwaltungsfunktionen Systemdaten lesen/schreiben Globale Variablen lesen/schreiben Programme lesen/schreiben	Systemdaten lesen/schreiben	Programmverwaltungsfunktionen Systemdaten lesen/schreiben Globale Variablen lesen/schreiben Programme lesen/schreiben
ROM	Die Programme bleiben sowohl beim Ausschalten der Versorgungsspannung als auch bei entladener Batterie erhalten. Es ist nur ein Lesezugriff möglich.		(Programmverwaltungsfunktionen gesperrt) Globale Variablen lesen/schreiben Programme lesen/schreiben	

Tab. 9-38: *Speicherzugriff und Parameter ROMDRV*

HINWEISE

Speichern Sie zur Sicherung der Systemdaten alle Parameter und Fehler-Logfiles. Das gilt auch für die Dateien, auf die ein Schreib- bzw. Lesezugriff durch die Programme (Befehle Open, Print und Input) erfolgt.

Unter Programmverwaltungsfunktionen versteht man u. a. das Kopieren, Löschen oder Umbenennen von Programmen im Steuergerät mit Hilfe der Teaching Box oder der Software RT ToolBox2.

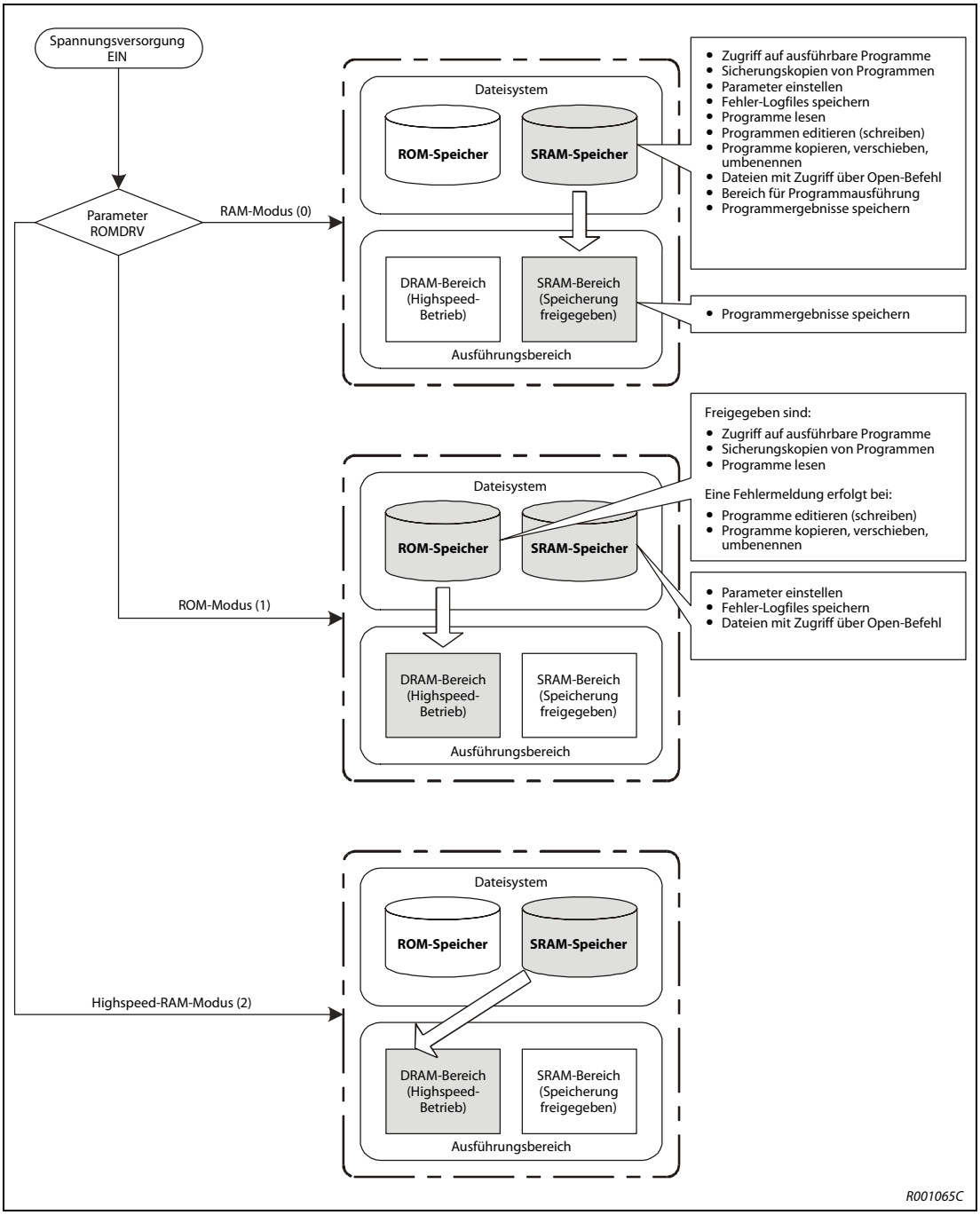


Abb. 9-24: *Speicherbereiche*

Der DRAM-Bereich dient im ROM- und Highspeed-RAM-Modus als Ausführungsspeicher. Die Verarbeitungsgeschwindigkeit ist etwa 1,2-mal so hoch wie die im SRAM-Bereich, der im einfachen RAM-Modus als Ausführungsspeicher dient. Die Verarbeitungsgeschwindigkeit hängt auch vom Programminhalt ab.

Im ROM- und Highspeed-RAM-Modus können alle Funktionen wie z. B. Programmausführung und Schrittbetrieb wie im RAM-Modus ausgeführt werden. Folgende Einschränkungen sind jedoch zu beachten:

- Variablen
  Im ROM-Modus und Highspeed-RAM-Modus können Variablen durch ausgeführte Programme geändert werden. Die geänderten Werte werden jedoch nach Ausschalten der Spannungsversorgung nicht gespeichert.

Variablen ①	**RAM-Modus**	**ROM-Modus**	**Highspeed-RAM-Modus**
Lokale Variablen	Die Variablenwerte bleiben auch nach Ausschalten der Spannungsversorgung gespeichert. ②	Während der Programmausführung werden die Werte lokaler Variablen gespeichert. Bei Umschaltung der Programme über das Bedienfeld oder externe E/A-Signale und beim Ausschalten der Spannungsversorgung werden die Werte gelöscht. Die Variablenwerte in einem Programm, dass über die CallP-Anweisung aufgerufen wurde, werden beim Rücksprung in das aufrufende Programm gelöscht.	Variablenwerte in einem Programm, dass beim Ausschalten der Spannungsversorgung ausgeführt wurde, werden gelöscht. Sie werden bei Auswahl eines Programms und nach Ausführung der CallP-Anweisung gespeichert.
Programmexterne Variablen	Die Variablenwerte bleiben auch nach Ausschalten der Spannungsversorgung gespeichert.	Die Variablenwerte bleiben bis zum Ausschalten der Spannungsversorgung gespeichert. (Sie werden bei Umschaltung der Programme nicht gelöscht. Änderungen der Variablenwerte gehen beim Ausschalten der Spannungsversorgung verloren.)	Die Variablenwerte bleiben auch nach Ausschalten der Spannungsversorgung gespeichert.
Benutzerdefinierte externe Variablen			Änderungen der Variablenwerte gehen beim Ausschalten der Spannungsversorgung verloren.

Tab. 9-39: *Variablen im RAM-, ROM und Highspeed-RAM-Modus*

① Die Variablentypen sind: numerische Variablen, Zeichenkettenvariablen, Positions- und Gelenkvariablen.

② Beim Überschreiben eines Programms mit Hilfe der Programmier-Software werden die im Programm verwendeten Werte aller lokalen Variablen gelöscht.

ACHTUNG:

Beim Highspeed-RAM-Modus und RAM-Modus bleiben die programmexternen Variablen auch nach Ausschalten der Spannungsversorgung gespeichert. Beachten Sie allerdings, dass die Variablenwerten nicht gespeichert werden, wenn die Spannungsversorgung direkt nach einer Änderung des Parameters ROMDRV abgeschaltet wird.

- Änderung von Variablen während der Programmausführung
  Werden Variablenwerte beim Abbruch eines Programms im ROM-Modus mit Hilfe der Teaching Box im Menü „Monitor-Funktion für Variable“ (siehe auch Abschn. 3.17.5) geändert, ist keine Fortsetzung des Programms möglich. Auch wenn die STOP-LED am Steuergerät leuchtet, startet das Programm bei einer erneuten Ausführung in der ersten Zeile. Achten Sie daher darauf, dass es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommen kann.
  Im ROM-Modus können die Variablen nicht über die „Monitor-Funktion für Variable“ geändert werden. Eine Änderung von Variablenwerten in Programmplätzen, die nicht editiert werden, kann über die Programmier-Software erfolgen.

- Programme
  Die Editierung von Programmen erfolgt im ROM-Bereich. Die im Steuergerät gespeicherten Programme sind geschützt (protect ON) und können im ROM-Modus nicht gelöscht werden. Bei Umschaltung in den RAM-Modus nimmt das Programm wieder den Status an, den es im RAM-Modus vor der Umschaltung hatte. Programme können im ROM-Betrieb gelesen aber nicht geschrieben werden. Dementsprechend kann ein Programm nicht kopiert oder umbenannt werden.

- Parameter
  Parameter und Fehler-Logfiles werden unabhängig vom Modus immer im RAM-Bereich gespeichert. Eine Änderung des Parameters RLNG (siehe auch Seite 9-30) zur Umschaltung der Programmiermethode im ROM-Betrieb ist nicht möglich. (Prüfen Sie vor Einstellung des Parameters im Technischen Handbuch, ob der von Ihnen verwendete Roboter über diese Funktion verfügt.)

- Backup
  Im ROM-Modus werden Programme aus dem ROM-Speicher und Parameter und Fehler-Logfiles aus dem RAM-Bereich gesichert.

- Direkte Befehlsausführung
  Im ROM-Modus können lokale Variable nicht durch direkte Befehlsausführung geschrieben werden.

- CTN-Funktion
  Im ROM-Modus ist die CTN-Funktion „Programm fortsetzen“ unwirksam, auch wenn sie aktiviert ist.

- Erweiterungsspeicher
  Wird im ROM-Modus ein Erweiterungsspeicher installiert, erfolgt eine Fehlermeldung. Die Installation eines Erweiterungsspeichers ist nur im RAM-Modus möglich.

- Einschaltzeit
  Die Einschaltzeit und die verbleibende Lebensdauer der Batterie werden unabhängig von der Umschaltung zwischen ROM- und RAM-Modus gezählt.

- Betriebsdaten
  Die Überwachung der Betriebsdaten (Programmzähler, Zykluszeit usw.) durch die Programmier-Software werden im ROM-Modus nicht aktualisiert.

9.21.2 Umschaltung zwischen ROM- und RAM-Modus

Folgendes Flussdiagramm zeigt die Vorgehensweise zum Umschalten vom ROM- in den RAM- und den Highspeed-RAM-Modus und umgekehrt.

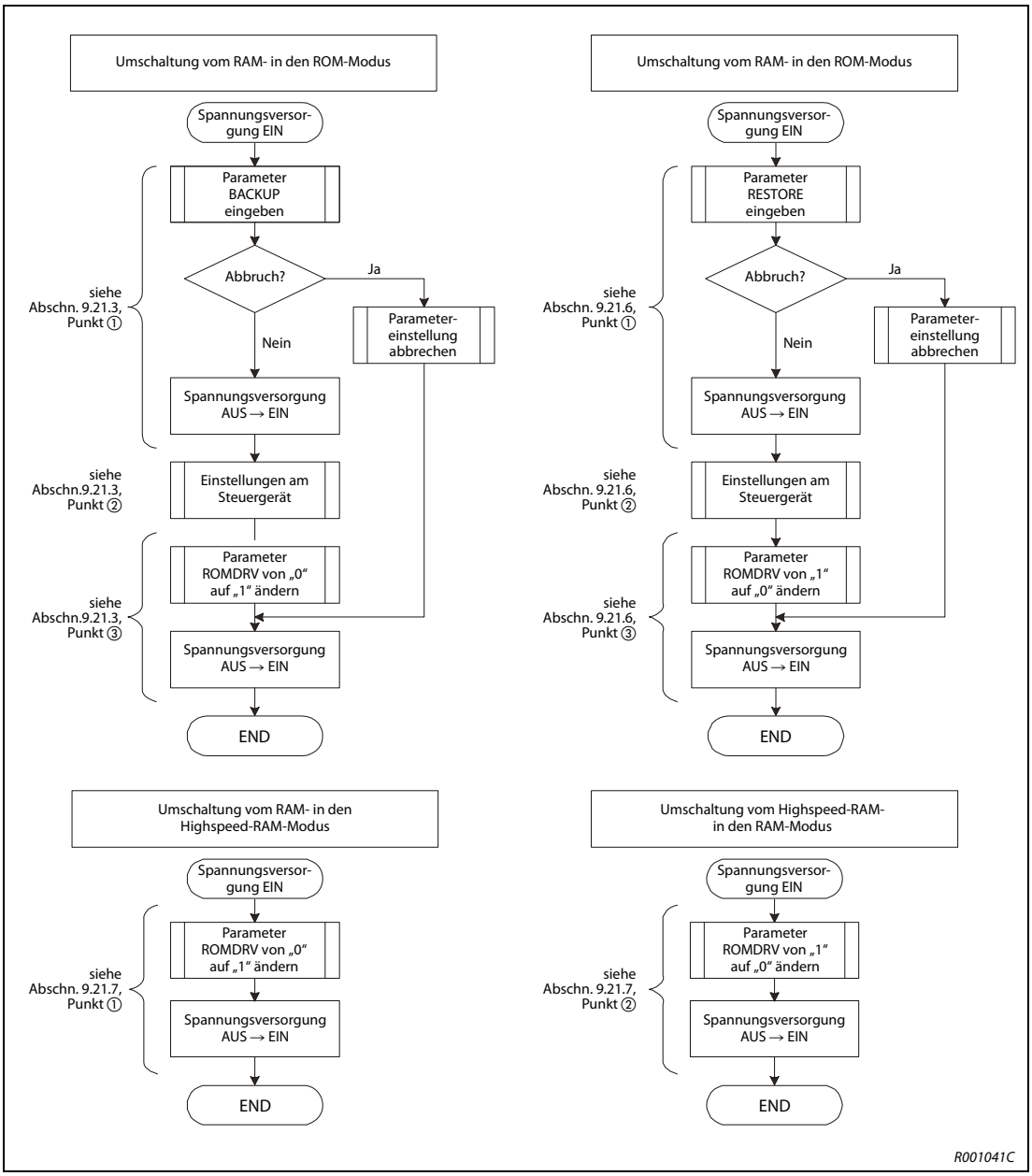


Abb. 9-25: *Umschaltung zwischen ROM- und RAM-Modus*

9.21.3 Umschaltung in den ROM-Modus

Gehen Sie zur Umschaltung in den ROM-Modus wie in den Schritten ① bis ③ beschrieben vor.

① Bereiten Sie den Kopiervorgang zur Übertragung der Daten vom RAM- in den ROM-Speicher vor. Die Programme, die vor der Umschaltung vom RAM- in den ROM-Modus erstellt wurden, sind im RAM-Speicher des Dateisystem im Steuergerät gespeichert. Nachdem die Programme im ROM-Speicher gelöscht sind, werden die Programme aus dem RAM-Speicher übertragen.

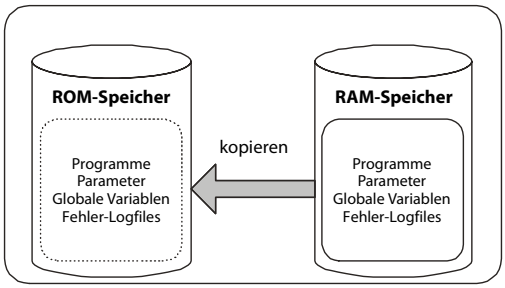
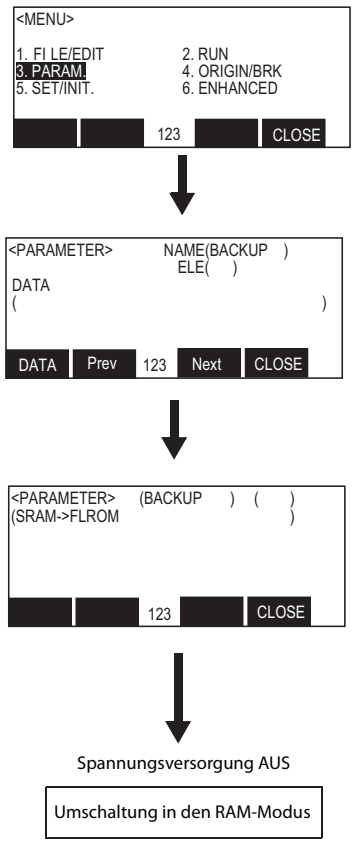


Abb. 9-26:
Dateisystem im Steuergerät

R001042C



1) Rufen Sie das Menü PARAMETER im Menü MAINTENANCE auf.

2) Geben Sie die Zeichen „BACKUP" ein und bestätigen Sie die Eingabe mit der [EXE]-Taste.

3) Erscheint die Anzeige „SRAM → FLROM " betätigen Sie erneut die [EXE]-Taste. Der Inhalt des Datenfeldes darf nicht geändert werden.
Es erfolgt nun ein Selbsttest, ob die Programme in den ROM-Speicher übertragen werden können. Ein Programm mit der Startbedingung „ALWAYS" wird während des Selbsttests automatisch unterbrochen. Nach Abschluss des Selbsttests springt der Cursor in das Feld zur Eingabe des Parameternamens. Treten während des Selbsttests Unregelmäßigkeiten auf, erfolgt die Ausgabe eine Fehlermeldung. Das Datum des Kopiervorgangs wird in einem Fehler-Logfile gespeichert.

Abbruch
Wenn die zum Menü „CLOSE" korrespondierende Funktionstaste betätigt wird, erfolgt keine Änderung der Daten.

4) Schalten Sie nun die Spannungsversorgung aus. Die Spannungsversorgung muss auch beim Abbruchvorgang unter 3) ausgeschaltet werden. Beim nächsten Einschaltvorgang erfolgt die Übertragung der Daten in den ROM-Speicher. Wird die Spannungsversorgung nach Ausführung von Schritt 3) nicht ausgeschaltet, erfolgt keine korrekte Übertragung der Daten in den ROM-Bereich.

5) Spannungsversorgung wieder einschalten
Nach dem Einschalten der Spannungsversorg erfolgt auf dem Steuergerät im Anzeigefeld „STATUS.NUMBER" die Anzeige „OK". Betätigen Sie nach der Anzeige „OK" die [START]-Taste am Steuergerät. (Der Vorgang wird auf der nächsten Seite detailliert beschrieben.

R001713E

Abb. 9-27: *Umschaltung in den RAM-Modus*

② Ausführung des Kopiervorgangs über das Steuergerät

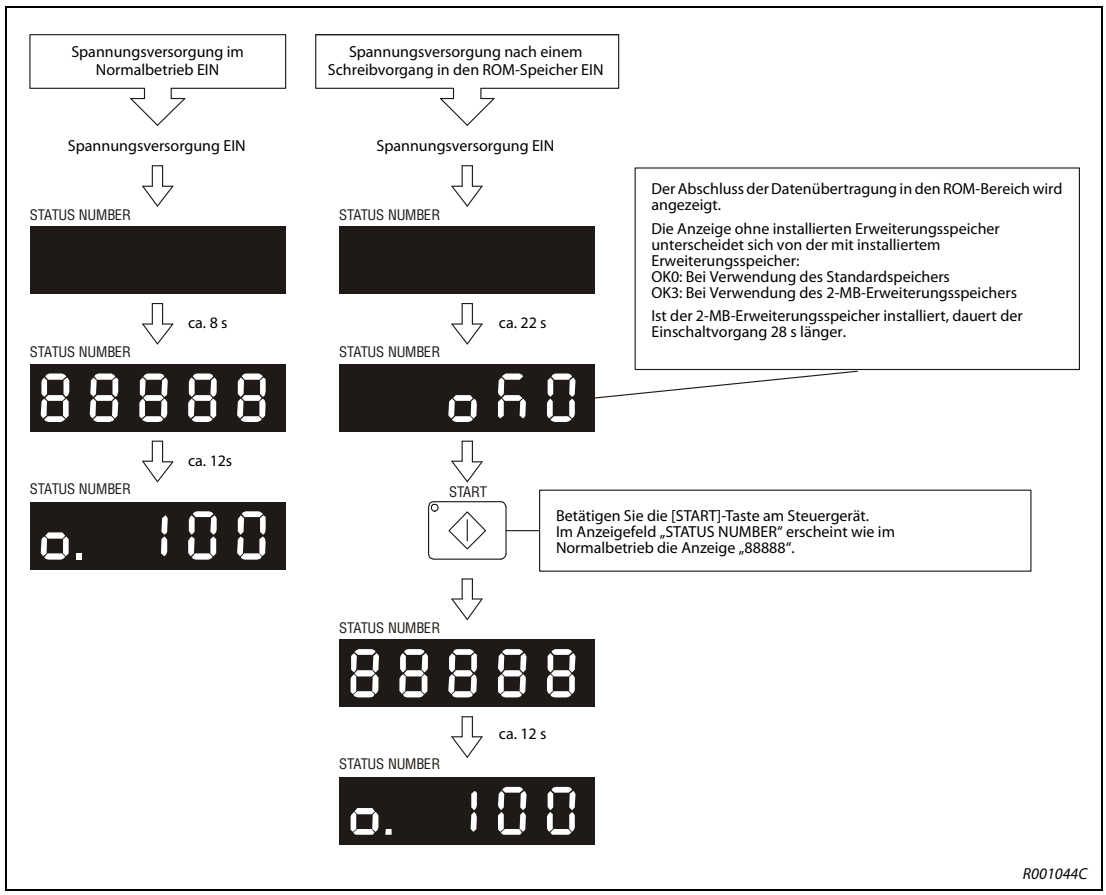
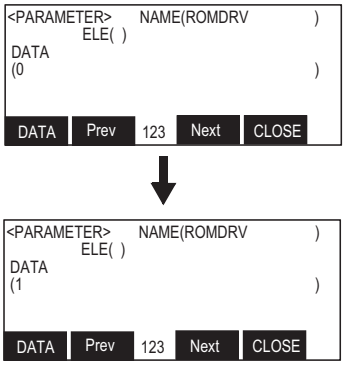

Spannungsversorgung im Normalbetrieb EIN
Spannungsversorgung nach einem Schreibvorgang in den ROM-Speicher EIN
Spannungsversorgung EIN
Spannungsversorgung EIN
STATUS NUMBER
STATUS NUMBER
ca. 8 s
ca. 22 s
STATUS NUMBER
STATUS NUMBER
ca. 12s
START
STATUS NUMBER
ca. 12 s
STATUS NUMBER
Der Abschluss der Datenübertragung in den ROM-Bereich wird angezeigt.
Die Anzeige ohne installierten Erweiterungsspeicher unterscheidet sich von der mit installiertem Erweiterungsspeicher:
OK0: Bei Verwendung des Standardspeichers
OK3: Bei Verwendung des 2-MB-Erweiterungsspeichers
Ist der 2-MB-Erweiterungsspeicher installiert, dauert der Einschaltvorgang 28 s länger.
Betätigen Sie die [START]-Taste am Steuergerät.
Im Anzeigefeld „STATUS NUMBER" erscheint wie im Normalbetrieb die Anzeige „88888".
R001044C

Abb. 9-28: *Ausführung des Kopiervorgangs*

ACHTUNG:

Erfolgt die Umschaltung vom RAM- in den ROM-Modus, ohne dass ein Programm in den ROM-Speicher kopiert wird, ist entweder kein Programm zur Ausführung im Speicher oder ein Programm, dessen Änderungen nicht geprüft sind, wird ausgeführt. Führen Sie daher unbedingt den oben beschriebenen Kopiervorgang aus.

③ Änderung des Parameterwertes und Umschaltung in den RAM-Modus



Ändern Sie die Einstellung des Parameters ROMDRV von „0" auf „1".

Bestätigen Sie die Eingabe durch Betätigung der [EXE]-Taste.
Schalten Sie anschließend die Spannungsversorgung aus und wieder ein.

R001714E

Abb. 9-29: *Parameter ROMDRV einstellen*

9.21.4 Anzeigen im ROM-Modus

Im ROM-Modus leuchtet in der rechten unteren Ecke der Anzeige „STATUS NUMBER“ ein Punkt.

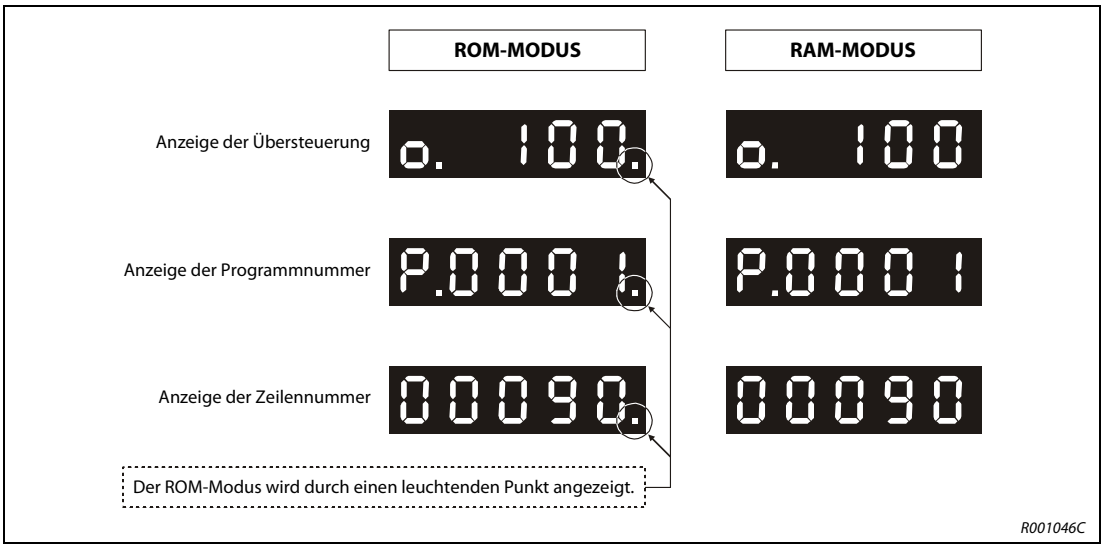


Abb. 9-30: *Anzeigen im ROM- und RAM-Modus*

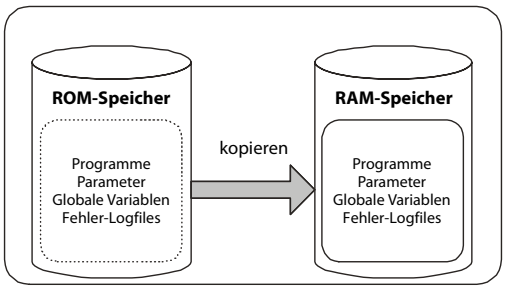
9.21.5 Programmeditierung im ROM-Modus

Während des Betriebs im ROM-Modus können Programme im Steuergerät gelesen werden. Bei Überschreiben eines Programmes erfolgt eine Fehlermeldung. Zur Programmeditierung ist zuerst der ROM-Modus zu unterbrechen (Änderung des Parameters ROMDRV von „1“ auf „0“ und Aus- und Wiedereinschalten der Spannungsversorgung). Dann kann des Programm editiert werden. Soll nach Abschluss der Programmeditierung zurück in den ROM-Modus geschaltet werden, stellen Sie sicher, dass Sie die Programme wieder in den ROM-Speicher kopieren.

9.21.6 Umschaltung in den RAM-Modus

Gehen Sie zur Umschaltung in den RAM-Modus wie in den Schritten ① bis ③ beschrieben vor.

① Bereiten Sie den Kopiervorgang zur Übertragung der Daten vom ROM- in den RAM-Speicher vor. Die Programme und Parameter, die bei der Umschaltung vom RAM- in den ROM-Modus in den ROM-Speicher kopiert wurden, werden zurück in den RAM-Speicher übertragen. Werden die Daten (Programme, Parameter, globale Variablen und Fehler-Logfiles) im RAM-Speicher gelöscht, erfolgt die Wiederherstellung aus dem ROM-Speicher.

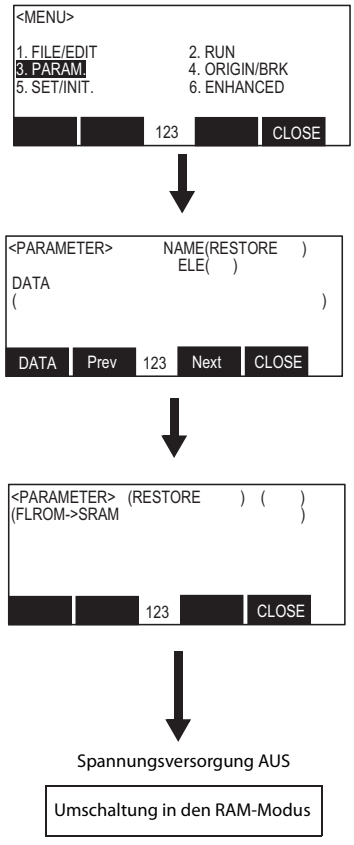


R001047C

Abb. 9-31: *Dateisystem im Steuergerät*

ACHTUNG:

Bei Wiederherstellung von Daten des RAM-Speichers im ROM-Bereich werden alle im ROM-Modus geänderten Werte der Parameter, globalen Variablen und Fehler-Logfiles überschrieben. Parameter, die im ROM-Modus geändert wurden, müssen vor der Umschaltung in den RAM-Modus erneut eingestellt werden.



1) Rufen Sie das Menü PARAMETER im Menü MAINTENANCE auf.

2) Geben Sie die Zeichen „RESTORE“ ein und bestätigen Sie die Eingabe mit der [EXE]-Taste.

3) Erscheint die Anzeige „FLROM → SRAM“ betätigen Sie erneut die [EXE]-Taste. Der Inhalt des Datenfeldes darf nicht geändert werden. Bereiten Sie die Wiederherstellung der Daten im RAM-Speicher vor. Ein Programm mit der Startbedingung „ALWAYS“ wird nicht unterbrochen. Nach Abschluss der Vorbereitungsphase springt der Cursor in das Feld zur Eingabe des Parameternamens.

   Abbruch
   Wenn die zum Menü „CLOSE“ korrespondierende Funktionstaste betätigt wird, erfolgt keine Änderung der Daten.

4) Schalten Sie nun die Spannungsversorgung aus. Die Spannungsversorgung muss auch beim Abbruchvorgang unter 3) ausgeschaltet werden. Beim nächsten Einschaltvorgang erfolgt die Übertragung der Daten in den RAM-Speicher. Wird die Spannungsversorgung nach Ausführung von Schritt 3) nicht ausgeschaltet, erfolgt keine korrekte Übertragung der Daten in den RAM-Bereich.

5) Spannungsversorgung wieder einschalten
   Nach dem Einschalten der Spannungsversorg erfolgt auf dem Steuergerät im Anzeigefeld „STATUS.NUMBER“ die Anzeige „OK“. Betätigen Sie nach der Anzeige „OK“ die [START]-Taste am Steuergerät. (Der Vorgang wird auf der nächsten Seite detailliert beschrieben.)

R001715E

Abb. 9-32: *Umschaltung in den RAM-Modus*

② Wiederherstellung der Daten über das Steuergerät

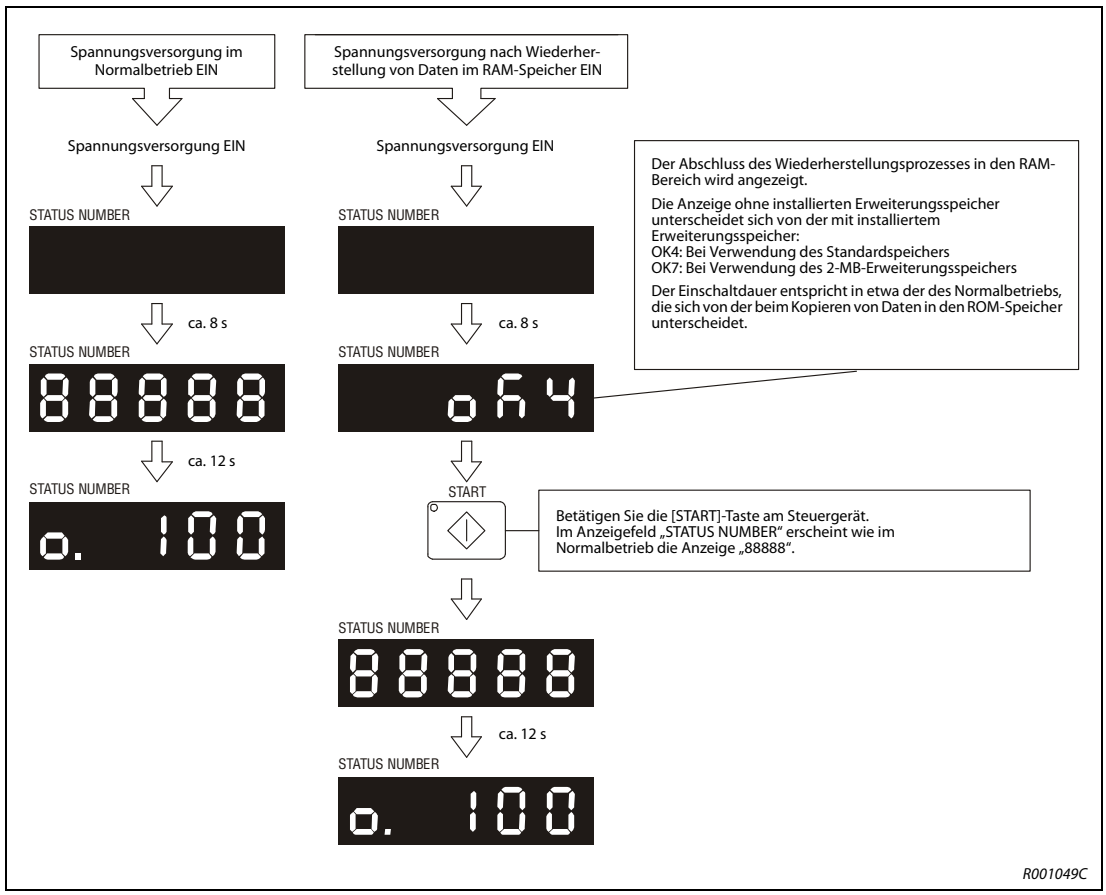
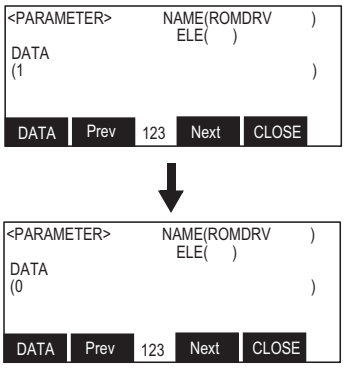

Spannungsversorgung im Normalbetrieb EIN
Spannungsversorgung nach Wiederherstellung von Daten im RAM-Speicher EIN
Spannungsversorgung EIN
Spannungsversorgung EIN
STATUS NUMBER
STATUS NUMBER
ca. 8 s
ca. 8 s
STATUS NUMBER
STATUS NUMBER
88888
oK4
ca. 12 s
START
STATUS NUMBER
o. 100
Der Abschluss des Wiederherstellungsprozesses in den RAM-Bereich wird angezeigt.
Die Anzeige ohne installierten Erweiterungsspeicher unterscheidet sich von der mit installiertem Erweiterungsspeicher:
OK4: Bei Verwendung des Standardspeichers
OK7: Bei Verwendung des 2-MB-Erweiterungsspeichers
Der Einschaltdauer entspricht in etwa der des Normalbetriebs, die sich von der beim Kopieren von Daten in den ROM-Speicher unterscheidet.
Betätigen Sie die [START]-Taste am Steuergerät.
Im Anzeigefeld „STATUS NUMBER" erscheint wie im Normalbetrieb die Anzeige „88888".
STATUS NUMBER
88888
ca. 12 s
STATUS NUMBER
o. 100
R001049C

Abb. 9-33: *Wiederherstellung der Daten*

③ Änderung des Parameterwertes und Umschaltung in den RAM-Modus



Ändern Sie die Einstellung des Parameters ROMDRV von „1" auf „0".

Bestätigen Sie die Eingabe durch Betätigung der [EXE]-Taste.

Schalten Sie anschließend die Spannungsversorgung aus und wieder ein.

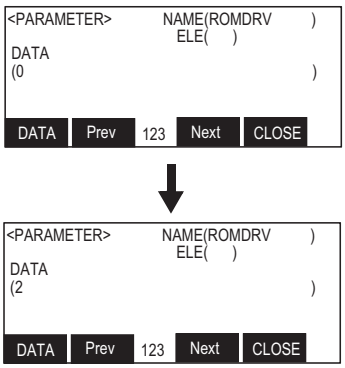
R001716E

Abb. 9-34: *Parameter ROMDRV einstellen*

9.21.7 Umschaltung in den Highspeed-RAM-Modus

Gehen Sie zur Umschaltung in den Highspeed-RAM-Modus und zurück in den RAM-Modus wie folgt vor.

① Änderung des Parameterwertes und Umschaltung in den Highspeed-RAM-Modus (DRAM)



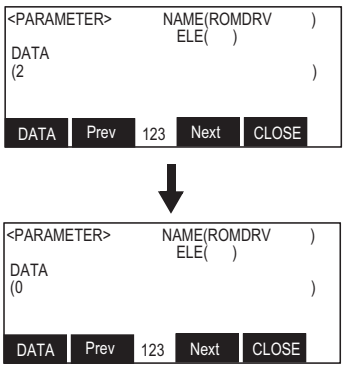
Ändern Sie die Einstellung des Parameters ROMDRV von „0“ auf „2“.

Bestätigen Sie die Eingabe durch Betätigung der [EXE]-Taste.
Schalten Sie anschließend die Spannungsversorgung aus und wieder ein.

R001717E

Abb. 9-35: *Parameter ROMDRV einstellen*

② Änderung des Parameterwertes und Umschaltung zurück in den RAM-Modus



Ändern Sie die Einstellung des Parameters ROMDRV von „2“ auf „0“.

Bestätigen Sie die Eingabe durch Betätigung der [EXE]-Taste.
Schalten Sie anschließend die Spannungsversorgung aus und wieder ein.

R001718E

Abb. 9-36: *Parameter ROMDRV einstellen*

9.22 Warmlaufbetrieb

9.22.1 Funktionsbeschreibung

Das Beschleunigungs- und Bremsverhalten sowie das Servosystem der MITSUBISHI-Roboter sind so ausgelegt, dass der Roboter in Umgebungen mit normalen Temperaturbedingungen die bestmöglichen Betriebswerte erreicht. Niedrige Temperaturen oder längere Betriebspausen beeinflussen die Viskosität der verwendeten Schmiermittel. Dadurch kann die Präzision abnehmen und es können Servofehler auftreten, die zu Positionsabweichungen führen. In einem solchen Fall empfiehlt es sich, den Roboter erst nach einer Anwärmphase mit niedriger Geschwindigkeit in den endgültigen, zeitoptimierten Produktionsprozess zu integrieren. Dazu wäre jedoch ein separates Programm für den Betrieb in der Anwärmphase erforderlich.

Im Warmlaufbetrieb wird der Roboter direkt nach dem Einschalten des Steuergerätes mit verminderter Geschwindigkeit verfahren. Anschließend erfolgt eine kontinuierliche Anhebung der Geschwindigkeit, die nach Ablauf der für den Warmlaufbetrieb festgelegten Zeit ihren Endwert erreicht. Der Warmlaufbetrieb erlaubt die Ausführung einer Anwärmphase, ohne dass ein separates Programm erstellt werden muss. Treten beim Einsatz des Roboters in Umgebungen mit niedrigen Temperaturen oder nach längeren Betriebspausen Positionsabweichungen auf, aktivieren Sie den Warmlaufbetrieb.

9.22.2 Aktivierung des Warmlaufbetriebs

Setzen Sie den Parameter WUPENA auf „1“, um den Warmlaufbetrieb freizugeben. Schalten Sie anschließend die Spannungsversorgung des Steuergerätes aus und wieder ein.

HINWEIS Um die Funktion des Warmlaufbetriebs mit einem Roboter zu verwenden, der nicht zur RV-SD-Serie gehört, muss – anders als bei der Aktivierung über den Parameter WUPENA – mit Hilfe des Parameters WUPAXIS eine Achse für den Warmlaufbetrieb festgelegt werden. Eine detaillierte Beschreibung finden Sie in Abschn. 9.22.4.

9.22.3 Aktivierter Warmlaufbetrieb

Ist der Warmlaufbetrieb freigegeben, wird er durch Ein- und Wiederausschalten der Spannungsversorgung des Steuergerätes aktiviert. (Die Geschwindigkeit wird automatisch reduziert.)

Im Warmlaufbetrieb bewegt der Roboter sich mit einer Geschwindigkeit, die kleiner als die festgelegte Betriebsgeschwindigkeit ist. Anschließend steigt die Geschwindigkeit wieder, bis sie nach Ablauf der für die Achse festgelegten Zeit für den Warmlaufbetrieb ihren Endwert erreicht.

Die Steilheit der Geschwindigkeitsänderung wird über die Übersteuerung im Warmlaufbetrieb festgelegt. Bei einem Wert von 100 % entspricht die Geschwindigkeit des Roboters der festgelegten Betriebsgeschwindigkeit. In der werksseitigen Parametereinstellung ändert sich die Übersteuerung in der für die Achse festgelegten Zeit für den Warmlaufbetrieb (siehe folgende Abbildung).

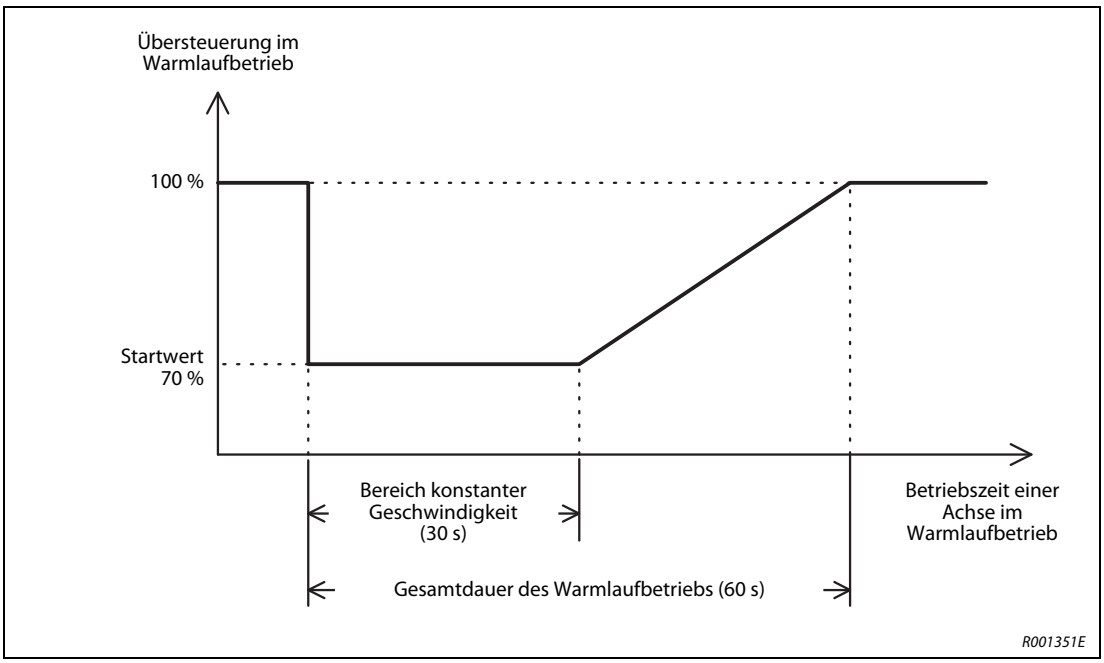


Abb. 9-37: *Änderungen während des Warmlaufbetriebs*

ACHTUNG:

- ***Ist der [MODE]-Umschalter des Steuergerätes auf „TEACH" eingestellt, im JOG-Betrieb oder im externen Echtzeit-Steuerungs-Betrieb (Mxt-Befehl) bewegt sich der Roboter auch bei aktiviertem Warmlaufbetrieb nicht mit der reduzierten Geschwindigkeit sondern mit der festgelegten Betriebsgeschwindigkeit.***
- ***Da sich der Roboter im Warmlaufbetrieb mit einer kleineren Geschwindigkeit als der festgelegten Betriebsgeschwindigkeit bewegt, ist eine Synchronisierung mit umliegenden Einheiten vorzusehen.***
- ***Bei niedriger Betriebsbeanspruchung der für den Warmlaufbetrieb festgelegten Achse, können auch bei aktiviertem Warmlaufbetrieb Servofehler wie z. B. Positionsabweichungen auftreten. Ändern Sie in diesem Fall das Programm und vermindern Sie sowohl die Beschleunigungs-/Bremszeiten als auch die Geschwindigkeit.***

Der Warmlaufbetrieb wird bei Anzeige des Übersteuerungswerts auf der STATUS.NUMBER-Anzeige des Steuergeräts durch einen Unterstrich (_) an der zweiten Stelle gekennzeichnet.

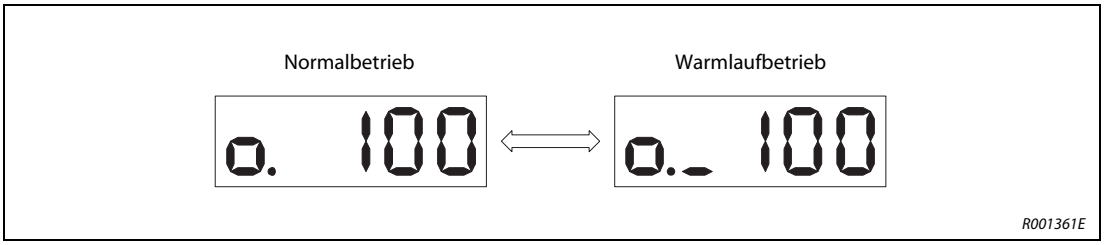


Abb. 9-38: *STATUS.NUMBER-Anzeige im Warmlaufbetrieb*

Bei Abbruch des Warmlaufbetriebs wird die für den Warmlaufbetrieb festgelegte Achse wieder mit der Betriebsgeschwindigkeit verfahren. Bei niedrigen Temperaturen kühlt sich der Gelenkbereich der Achse nach dem Abbruch des Warmlaufbetriebs in einer Betriebspause wieder ab. Nach längeren Betriebspausen der für den Warmlaufbetrieb festgelegten Achse (Werkseinstellung: 60 min) sollte der Warmlaufbetrieb erneut aktiviert und der Roboter mit einer reduzierten Geschwindigkeit verfahren werden.

HINWEISE

Dauert der ausgeschaltete Zustand beim Aus- und Wiedereinschalten des Steuergerätes nur kurze Zeit an, kühlt das Gelenk des Roboters auch nur wenig ab. Daher wird der Roboter nach Beendigung des Warmlaufbetriebs und einer kurzen Ausschaltzeit mit der Betriebsgeschwindigkeit und nicht im Warmlaufbetrieb verfahren.

Die Festlegung der Achse, die im Warmlaufbetrieb verfahren werden soll, erfolgt im Parameter WUPAXIS.

9.22.4 Parameter, spezielle Ein- und Ausgänge und Statusvariablen im Warmlaufbetrieb

Folgende Parameter, spezielle Ein- und Ausgänge und Statusvariablen sind im Warmlaufbetrieb zusätzlich gültig:

Parameter	Beschreibung
WUPENA	Freigabe des Warmlaufbetriebs 0: gesperrt / 1: freigegeben
WUPAXIS	Auswahl der Achse für den Warmlaufbetrieb durch Ein- und Ausschalten der Bits im Hexadezimalcode (J1, J2 ... mit dem niedrigsten Bit beginnend) Bit EIN: Achse auswählen / Bit AUS: Achse nicht auswählen
WUPTIME	Dauer des Warmlaufbetriebs in Minuten Legen Sie die Gesamtdauer im ersten Element und die Wiederholschwelle im zweiten Element fest. Gesamtdauer: Legen Sie die Dauer fest, in der die Achse im Warmlaufbetrieb mit reduzierter Geschwindigkeit verfahren wird. (Einstellbereich 0 bis 60) Wiederholschwelle: Legen Sie die Zeit fest, nachdem nach Beendigung eines Warmlaufbetriebs und einer kontinuierlichen Betriebspause einer Achse erneut ein Warmlaufbetrieb aktiviert wird (Einstellbereich: 1 bis 1440)
WUPOvrd	Einstellung der Geschwindigkeit im Warmlaufbetrieb Legen Sie den Startwert im ersten Element und den Bereich der konstanten Geschwindigkeit im zweiten Element in Prozent fest. Startwert: Startwert der Geschwindigkeitsübersteuerung im Warmlaufbetrieb (Einstellbereich 50 bis 100) Faktor für den Bereich konstanter Geschwindigkeit: Stellen Sie den Bereich der konstanten Geschwindigkeit in Bezug zur Gesamtdauer des Warmlaufbetriebs ein (Einstellbereich 0 bis 50)

Tab. 9-40: *Parameter für den Warmlaufbetrieb*

Parameter	Ein-/Ausgang	Beschreibung
MnWUPENA (n = 1 bis 3) (Betriebsrechte erforderlich)	Eingang	Freigabe des Warmlaufbetriebs eines Mechanismus (n: Mechanismusnummer)
	Ausgang	Ausgabe bei freigegebenem Warmlaufbetrieb (n: Mechanismusnummer)
MnWUPMD (n = 1 bis 3)	Ausgang	Ausgabe bei aktiviertem Warmlaufbetrieb, in dem der Roboter mit reduzierter Geschwindigkeit verfahren wird (n: Mechanismusnummer)

Tab. 9-41: *Spezielle Ein- und Ausgänge im Warmlaufbetrieb*

Parameter	Beschreibung
M_Wupov	Enthält den Übersteuerungswert im Warmlaufbetrieb, der den Geschwindigkeitssollwert auf die Geschwindigkeit reduziert, mit der der Roboter im Warmlaufbetrieb verfahren wird.
M_Wuprt	Enthält die Restzeit, in der eine Achse im Warmlaufbetrieb verfahren wird, bis der Warmlaufbetrieb beendet ist.
M_Wupst	Enthält die Zeit bis nach Beendigung eines Warmlaufbetriebs ein erneuter Warmlaufbetrieb aktiviert wird.

Tab. 9-42: *Statusvariablen für den Warmlaufbetrieb*

9.22.5 Ausführung des Warmlaufbetriebs

Der Warmlaufbetrieb kann über Parameter oder ein spezielles Eingangssignal aktiviert werden.

- Aktivierung über Parameter

  Aktivieren Sie den Warmlaufbetrieb, indem Sie den Parameter WUPENA auf „1“ setzen. Schalten Sie anschließend das Steuergerät aus und wieder ein. In folgenden Fällen wird der Warmlaufbetrieb nicht aktiv, auch wenn er über den Parameter WUPENA freigegeben ist:

  – Wenn der Parameter WUPAXIS auf „0“ gesetzt ist. (Es ist keine Achse für den Warmlaufbetrieb festgelegt.)

  – Wenn das erste Element des Parameters WUPTIME auf „0“ gesetzt ist. (Die Gesamtdauer des Warmlaufbetrieb ist auf „0“ gesetzt.)

  – Wenn das erste Element des Parameters WUPOvrd auf „100“ gesetzt ist. (Die Geschwindigkeit wird im Warmlaufbetrieb nicht reduziert.)

Setzen Sie die Parameter zur Aktivierung des Warmlaufbetriebs auf die entsprechenden Werte.

HINWEIS

Bei allen Robotern, die nicht zur RV-SD-Serie gehören, ist der Parameter WUPAXIS werksseitig auf „0“ gesetzt. Zur Ausführung der Funktion, ist eine Achse für den Warmlaufbetrieb festzulegen. (Zielachse, die im Warmlaufbetrieb gesteuert wird; z. B. eine Achse, die bei niedrigen Temperaturen zu Positionsabweichungen führt.)

- Aktivierung über spezielles Eingangssignal

  Wird der Warmlaufbetrieb über das spezielle Eingangssignal MnWUPENA (n = 1 bis 3: Mechanismusnummer) freigegeben, muss das Steuergerät nicht aus- und wieder eingeschaltet werden. Das Ausgangssignal MnWUPENA (n = 1 bis 3: Mechanismusnummer) zeigt an, ob der Warmlaufbetrieb freigegeben ist.

HINWEISE

Um den Warmlaufbetrieb über das spezielle Eingangssignal freizugeben, müssen die zuvor beschriebenen Parameter entsprechend eingestellt werden.

Das spezielle Eingangssignal benötigt die Freigabe der Betriebsrechte für externe Signale. Während des Betriebs oder im JOG-Betrieb wird eine Eingabe des Signals nicht akzeptiert.

Der über das Eingangssignal festgelegte Status „freigegeben/gesperrt“ wird auch nach Entzug der Betriebsrechte weiter aufrechterhalten.

9.22.6 Wenn der Warmlaufbetrieb freigegeben ist

Ist der Warmlaufbetrieb freigegeben, wird er durch Aus- und Einschalten der Spannungsversorgung des Steuergerätes aktiviert.

Im aktivierten Warmlaufbetrieb wird der Roboter mit einer reduzierten Geschwindigkeit verfahren, die durch den Übersteuerungswert für den Warmlaufbetrieb festgelegt ist. Bis zum Ablauf der Gesamtdauer für den Warmlaufbetrieb erfolgt eine kontinuierliche Erhöhung der Geschwindigkeit bis auf den Wert der Betriebsgeschwindigkeit. Nach Beendigung des Warmlaufbetriebs wird der Roboter wieder mit der Betriebsgeschwindigkeit verfahren.

Zustand nach Einschalten der Spannungsversorgung des Steuergerätes

Ist der Warmlaufbetrieb freigegeben, wird er durch Aus- und Einschalten der Spannungsversorgung des Steuergerätes aktiviert.

Wird das Steuergerät nach Beendigung des Warmlaufbetriebs nur kurz aus- und dann wieder eingeschaltet, startet der Roboter im Normalbetrieb und nicht im Warmlaufbetrieb, da das Gelenk des Roboters während der kurzen ausgeschalteten Phase nicht stark abkühlen konnte. Genauer gesagt, startet der Roboter im Normalbetrieb, wenn folgende Bedingung erfüllt ist:

Nach Beendigung des Warmlaufbetriebs startet der Roboter dann im Normalbetrieb, wenn die Betriebspause der festgelegten Achse bis zum Einschalten des Steuergerätes kürzer als die im zweiten Element des Parameters WUPTIME eingestellten Zeit ist (Wiederholschwelle).

HINWEIS

Erfolgt die Freigabe des Warmlaufbetriebs über das spezielle Eingangssignal MnWUPENA (n = 1 bis 3: Mechanismusnummer), ist der Warmlaufbetrieb immer aktiviert.

Zustand des Warmlaufbetriebs

Ob sich der Roboter im Warmlauf- oder Normalbetrieb befindet, kann mit den folgenden drei Methoden überprüft werden:

- Abfrage über STATUS.NUMBER-Anzeige auf dem Steuergerät

  Der aktuelle Status wird auf der STATUS.NUMBER-Anzeige des Steuergerätes parallel zum Übersteuerungswert angezeigt. Der Warmlaufbetrieb ist durch einen Unterstrich (_) an der zweiten Stelle der Anzeige gekennzeichnet.

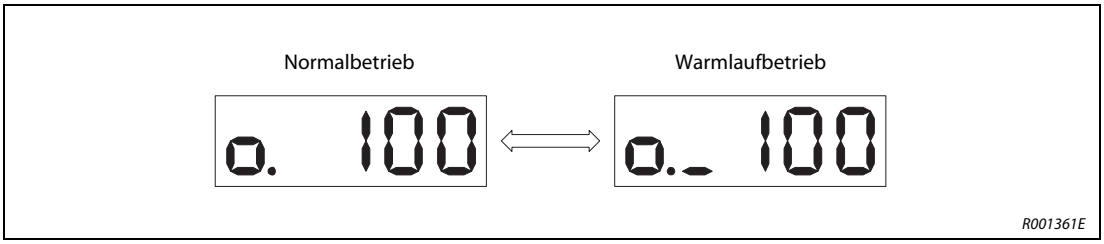


Abb. 9-39: *STATUS.NUMBER-Anzeige im Warmlaufbetrieb*

- Abfrage über Statusvariable

  Der aktuelle Status kann über den Wert der Stausvariablen M_Wupov abgefragt werden (Übersteuerungswert). Im Normalbetrieb ist der Wert der Stausvariablen 100 %, im Warmlaufbetrieb liegt er darunter.

- Abfrage über spezielles Ausgangssignal

  Im Warmlaufbetrieb wird das spezielle Ausgangssignal MnWUPMD (n = 1 bis 3: Mechanismusnummer) ausgegeben.

9.22.7 Umschaltung zwischen Normal- und Warmlaufbetrieb

Überschreitet die Betriebszeit einer Achse im Warmlaufbetrieb die Gesamtdauer für den Warmlaufbetrieb, wird der Warmlaufbetrieb beendet und es erfolgt eine Umschaltung auf den Normalbetrieb. Kühlt die Achse in einer längeren Betriebspause, deren Dauer den eingestellten Schwellwert für einen erneuten Warmlaufbetrieb überschreitet, wieder ab, so erfolgt eine Umschaltung vom Normal- in den Warmlaufbetrieb.

Beendigung des Warmlaufbetriebs

Überschreitet die Betriebszeit einer Achse im Warmlaufbetrieb die Gesamtdauer für den Warmlaufbetrieb, wird der Warmlaufbetrieb beendet und es erfolgt eine Umschaltung auf den Normalbetrieb. Legen Sie die Gesamtdauer für den Warmlaufbetrieb im ersten Element des Parameters WUPTIME fest. (Die Werkseinstellung ist 1 min.)

Werden mehrere Achsen im Warmlaufbetrieb verfahren, erfolgt die Beendigung des Warmlaufbetriebs, wenn die Betriebsdauer aller festgelegten Achsen die eingestellte Gesamtbetriebsdauer für den Warmlaufbetrieb überschreitet. Mit Hilfe der Statusvariablen M_Wuprt kann die Dauer einer gesteuerten Achse überprüft werden, in der die Achse noch im Warmlaufbetrieb verfahren wird.

Umschaltung vom Normalbetrieb in den Warnmeldungsbetrieb

Überschreitet die Betriebspause einer für den Warmlaufbetrieb festgelegten Achse den für einen erneuten Warmlaufbetrieb eingestellten Schwellwert, erfolgt eine Umschaltung vom Normalbetrieb in den Warmlaufbetrieb. Legen Sie den Schwellwert für einen erneuten Warmlaufbetrieb im zweiten Element des Parameters WUPTIME fest. (Die Werkseinstellung ist 6 min.)

Sind mehrere Achsen für den Warmlaufbetrieb festgelegt, startet der Warmlaufbetrieb, wenn eine der Achsen den Schwellwert für einen erneuten Warmlaufbetrieb überschreitet.
Mit Hilfe der Statusvariablen M_Wupst kann die Dauer einer gestoppten Achse bis zur Umschaltung in den Warmlaufbetrieb überprüft werden.

HINWEIS Wird eine für den Warmlaufbetrieb festgelegte Achse während des Roboterbetriebs nicht verfahren, so wird die Achse als gestoppt bewertet.

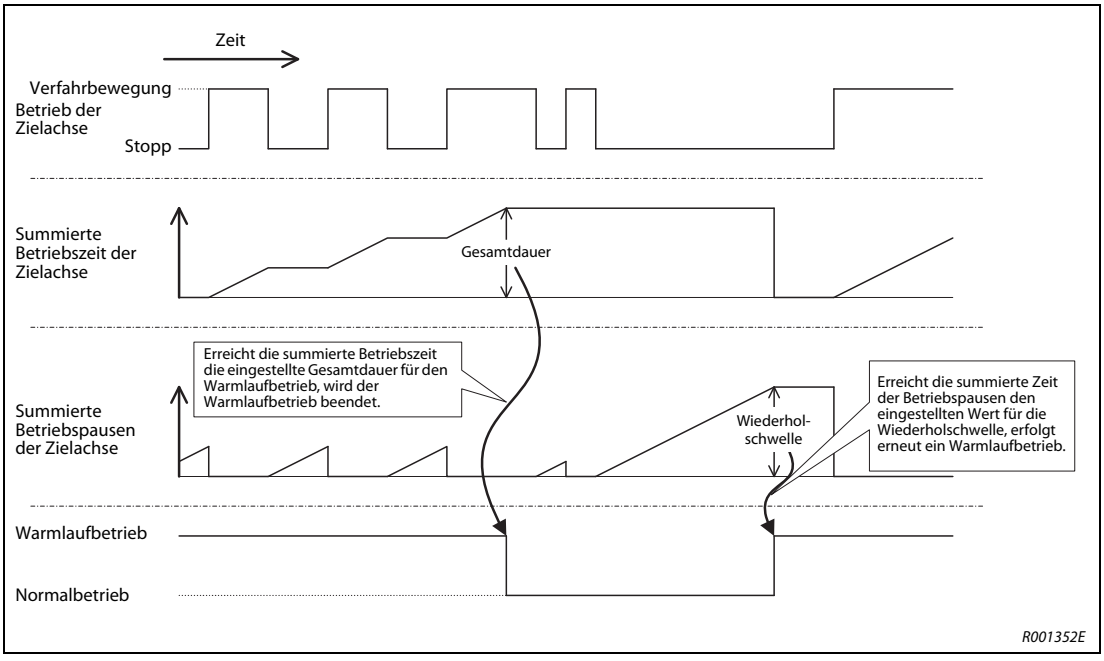


Abb. 9-40: *Beispiel für eine Umschaltung vom Normal- in den Warmlaufbetrieb*

Übersteuerungswert für den Warmlaufbetrieb

Die reduzierte Geschwindigkeit im Warmlaufbetrieb wird durch den Übersteuerungswert für den Warmlaufbetrieb bestimmt. Der Übersteuerungswert ändert sich in Abhängigkeit der Betriebsdauer einer Achse und beeinflusst direkt die Verfahrgeschwindigkeit des Roboters.

Legen Sie mit Hilfe des Parameters WUPOvrd den Startwert der Übersteuerung für den Warmlaufbetrieb und die Zeit in Bezug zur Gesamtdauer des Warmlaufbetriebs fest, in der die Übersteuerung konstant bleiben soll. (Die Werkseinstellungen sind 70 % für den Startwert und 50 % (= 30 s) für den Bereich der konstanten Geschwindigkeit.)

Mit Hilfe der Statusvariablen M_Wupov kann die Übersteuerung überprüft werden.

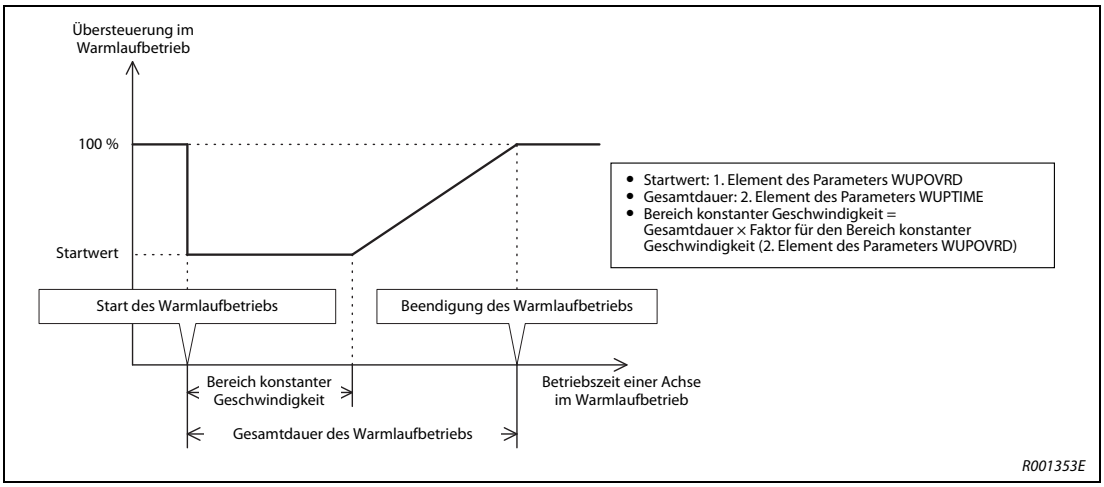

Übersteuerung im Warmlaufbetrieb
100 %
Startwert
Start des Warmlaufbetriebs
Beendigung des Warmlaufbetriebs
Bereich konstanter Geschwindigkeit
Gesamtdauer des Warmlaufbetriebs
Betriebszeit einer Achse im Warmlaufbetrieb
• Startwert: 1. Element des Parameters WUPOVRD
• Gesamtdauer: 2. Element des Parameters WUPTIME
• Bereich konstanter Geschwindigkeit = Gesamtdauer × Faktor für den Bereich konstanter Geschwindigkeit (2. Element des Parameters WUPOVRD)
R001353E

Abb. 9-41: *Übersteuerung im Warmlaufbetrieb*

Die aktuelle Übersteuerung im Warmlaufbetrieb ergibt sich wie folgt:

Gelenk-Interpolation	=	Übersteuerungswert der T/B oder des Steuergeräts	x	Einstellwert des Ovrd-Befehls	x	Einstellwert des JOvrd-Befehls	=	Übersteuerungswert für den Warmlaufbetrieb
Linear-Interpolation	=	Übersteuerungswert der T/B oder des Steuergeräts	x	Einstellwert des Ovrd-Befehls	x	Einstellwert des Spd-Befehls	=	Übersteuerungswert für den Warmlaufbetrieb

HINWEISE

Ist der [MODE]-Umschalter des Steuergerätes auf „TEACH" eingestellt, im JOG-Betrieb oder im externen Echtzeit-Steuerungs-Betrieb (Mxt-Befehl) ist der Übersteuerungswert für den Warmlaufbetrieb unwirksam und der Roboter wird auch bei aktiviertem Warmlaufbetrieb nicht mit der reduzierten Geschwindigkeit verfahren, sondern mit der festgelegten Betriebsgeschwindigkeit.

Da sich der Roboter im Warmlaufbetrieb mit einer kleineren Geschwindigkeit als der festgelegten Betriebsgeschwindigkeit bewegt, ist eine Synchronisierung mit umliegenden Einheiten vorzusehen.

Sind mehrere Achsen für den Warmlaufbetrieb festgelegt, erfolgt die Festlegung der Übersteuerung durch die Achse mit der kleinsten Betriebsdauer im Warmlaufbetrieb. Wird eine bestimmte, für den Warmlaufbetrieb festgelegte Achse nicht verfahren und der Wert der Statusvariablen M_Wuprt ändert sich nicht, ändert sich – unabhängig von den anderen Achsen, die verfahren werden – auch der Wert der Übersteuerung nicht.
In Abhängigkeit davon, ob eine Achse gesteuert oder gestoppt ist, kann sich der Übersteuerungswert wieder auf den Startwert ändern, bevor er 100 % erreicht hat.
Übersteigt z. B. der Übersteuerungswert für den Warmlaufbetrieb den Startwert und eine bestimmte Achse schaltet vom Normalbetrieb in den Warmlaufbetrieb, so ist diese Achse die mit der geringsten Betriebsdauer (Betriebsdauer = 0 s). Der Übersteuerungswert für den Warmlaufbetrieb wird wieder auf den Startwert gesetzt.

9.22.8 Alarme im Warmlaufbetrieb

Trotz Warmlaufbetrieb treten Positionsabweichungen auf

- Tritt der Fehler auf, wenn der Startwert im Warmlaufbetrieb wirksam ist, verkleinern Sie den Startwert (1. Element des Parameters WUPOvrd).
- Tritt der Fehler auf, wenn die Übersteuerung im Warmlaufbetrieb auf 100 % ansteigt, ist eventuell die Gesamtbetriebszeit für den Warmlaufbetrieb oder der Bereich der konstanten Geschwindigkeit zu klein. Vergrößern Sie in diesem Fall das erste Element (Gesamtbetriebszeit für den Warmlaufbetrieb) oder das zweite Element (Faktor für den Bereich konstanter Geschwindigkeit) des Parameters WUPOvrd.
- Kann der Fehler durch die oben genannten Maßnahmen nicht behoben werden, ändern Sie das Programm, verringern Sie die Geschwindigkeiten und die Einstellungen für die Beschleunigung-/ Abbremsung.

Nach Beendigung des Warmlaufbetriebs treten Positionsabweichungen auf

- Vergrößern Sie den Wert des ersten Elements im Parameter WUPTIME und verlängern Sie die Gesamtbetriebsdauer des Warmlaufbetriebs.
- Prüfen Sie, ob die Last des Roboters und die Umgebungstemperatur innerhalb der zulässigen Bereiche liegen.
- Prüfen Sie, ob eine Achse nach Beendigung des Warmlaufbetrieb für längere Zeit gestoppt ist. Erhöhen Sie in einem solchen Fall das zweite Element des Parameters WUPTIME und verkürzen Sie die Zeit bis zu einer erneuten Ausführung des Warmlaufbetriebs.
- Kann der Fehler durch die oben genannten Maßnahmen nicht behoben werden, ändern Sie das Programm, verringern Sie die Geschwindigkeiten und die Einstellungen für die Beschleunigung-/ Abbremsung.

Der Warmlaufbetrieb wird nicht beendet

- Prüfen Sie den Wert des Parameters WUPAXIS, um festzustellen, ob eine Achse, die nicht gesteuert wird, für den Warmlaufbetrieb festgelegt wurde.
- Prüfen Sie, ob die Betriebspause einer für den Warmlaufbetrieb festgelegten Achse den Wert für die Wiederholschwelle (2. Element des Parameters WUPTIME) übersteigt.
- Prüfen Sie, ob der Roboter mit einer extrem niedrigen Geschwindigkeit verfahren wird (3 % bis 5 % Übersteuerung bei Gelenkinterpolation). Bei dieser niedrigen Geschwindigkeit ist kein Warmlaufbetrieb nötig. Deaktivieren Sie den Warmlaufbetrieb.

9.23 Durchfahren eines singulären Punktes

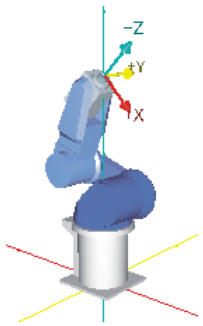
MITSUBISHI-Robotersysteme berechnen linear interpolierte Bewegungen und speichern geteachte Positionen als Positionsdaten im XYZ-Koordinatensystem. Für einen 6-achsigen Roboter können Positionen über die Koordinaten X, Y, Z, A, B, und C festgelegt werden. Es gibt jedoch eine Vielzahl von Positionen mit den gleichen Positionsdaten, aber mit unterschiedlichen Roboterstellungen (Stellung der Robotergelenke). Diese unterschiedlichen Roboterstellungen werden über sogenannte Stellungsmerker eindeutig festgelegt. Jedes einzelne Robotergelenk kann jedoch eine unendliche Anzahl von Winkeln einnehmen.

Auch unter Zuhilfenahme des Stellungsmerkers ist es an den Stellen, an denen der Stellungsmerker umschaltet, nicht immer möglich, den Roboter mit der gewünschten Position und Stellung zu steuern. (Bei einem 6-achsigen Knickarmroboter z. B. sind die Achsen J4 und J6 nicht eindeutig definiert, wenn der Winkel der J5-Achse 0° ist.) Diese Positionen werden singuläre Punkte genannt. Sie können im XYZ-JOG-Betrieb und mit Linear-Interpolation nicht erreicht werden. Früher wurde das Problem umgangen, indem man bei der Planung der Bewegungsabläufe singuläre Punkte vermied oder der Roboter bei einer unvermeidbaren Durchfahrt eines singulären Punktes mit Gelenk-Interpolation verfahren wurde.

Die Funktion zur Durchfahrt singulärer Punkte erlaubt ein Durchfahren singulärer Punkte im XYZ-JOG-Betrieb und mit Linear-Interpolation. Dadurch wird eine freiere Planung der Bewegungsabläufe und eine Vergrößerung des Arbeitsraumes durch die Nutzung der Linear-Interpolation möglich.

9.23.1 Positionen singulärer Punkte, die durchfahren werden können

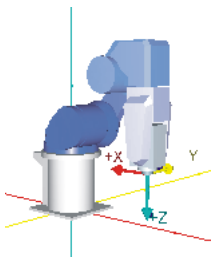
Folgende Positionen können mit der Funktion zur Durchfahrt singulärer Punkte durchfahren werden.



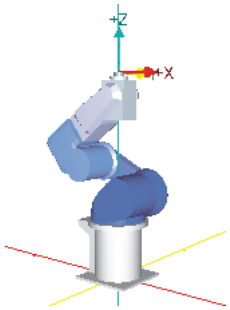
1. Positionen, in denen der Mittelpunkt des Handflansches mit der J1-Achse zusammenfällt.
   In dieser Position wechselt der Stellungsmerker zwischen den Stellungen „rechts“ und „links“.

Abb. 9-42:
Bei fünfachsigen Knickarmrobotern

R001354C



1. Positionen mit J5 = 0°
   In dieser Position wechselt der Stellungsmerker zwischen den Stellungen „kippen“ und „nicht kippen“.



2. Positionen, in denen der Mittelpunkt der J5-Achse mit der J1-Achse zusammenfällt
   In dieser Position wechselt der Stellungsmerker zwischen den Stellungen „rechts“ und „links“.

R001355C

Abb. 9-43: *Bei sechsachsigen Knickarmrobotern*

9.23.2 Betrieb mit aktivierter Funktion zum Durchfahren singulärer Punkte

Ist die Funktion zum Durchfahren singulärer Punkte aktiviert, kann der Roboter im XYZ-JOG-Betrieb oder mittels Linear-Interpolation von der Position A über die Position B (singulärer Punkt) zur Position C und umgekehrt verfahren werden. Der Stellungsmerker wechselt den Status vor und nach Durchfahren des Punktes B.

Ist die Funktion zum Durchfahren singulärer Punkte deaktiviert (oder sie wird nicht unterstützt), stoppt der Roboter vor der Ausführung der Bewegung von A nach B und es erfolgt die Ausgabe einer Fehlermeldung. Im XYZ-JOG-Betrieb stoppt der Roboter unmittelbar vor der Position B.

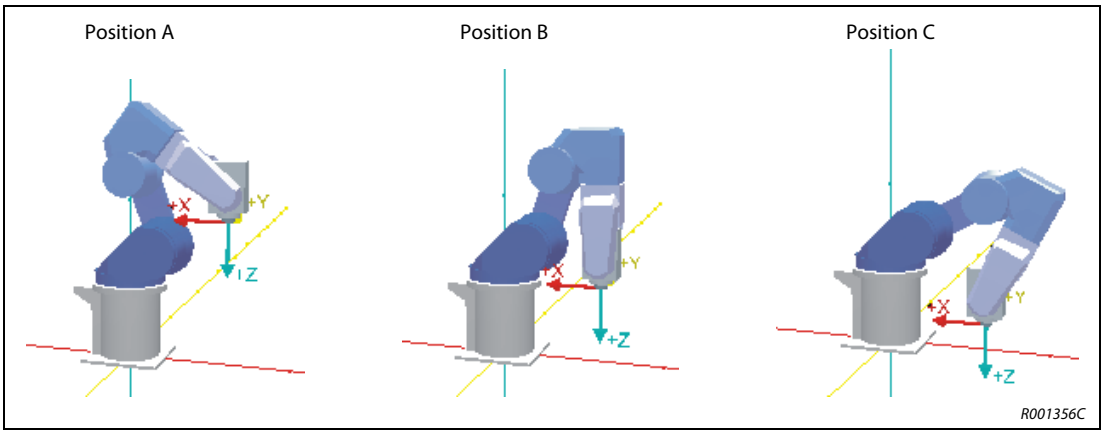


Abb. 9-44: *Durchfahren eines singulären Punktes*

Der Roboter kann einen singulären Punkt durchfahren, wenn der Verfahrweg durch den singulären Punkt verläuft. Verläuft der Verfahrweg nicht durch den singulären Punkt (sondern in der Nähe des singulären Punkts) bewegt sich der Roboter, ohne dass ein Zustandswechsel des Stellungsmerkers erfolgt.

In der folgenden Abbildung verläuft der Verfahrweg beim Durchfahren der Positionen D -> E -> F durch einen singulären Punkt. (Der Stellungsmerker wechselt den Status vor und nach Durchfahren des Punktes E.)

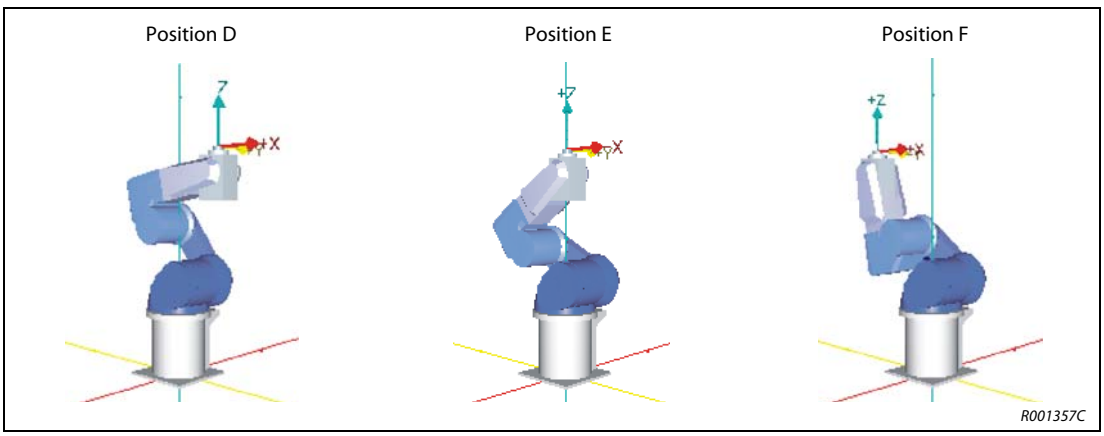


Abb. 9-45: *Durchfahren eines singulären Punktes*

In der folgenden Abbildung verläuft der Verfahrweg beim Durchfahren der Positionen G -> H -> I in der Nähe des singulären Punkts (Der Stellungsmerker wechselt den Zustand nicht.)

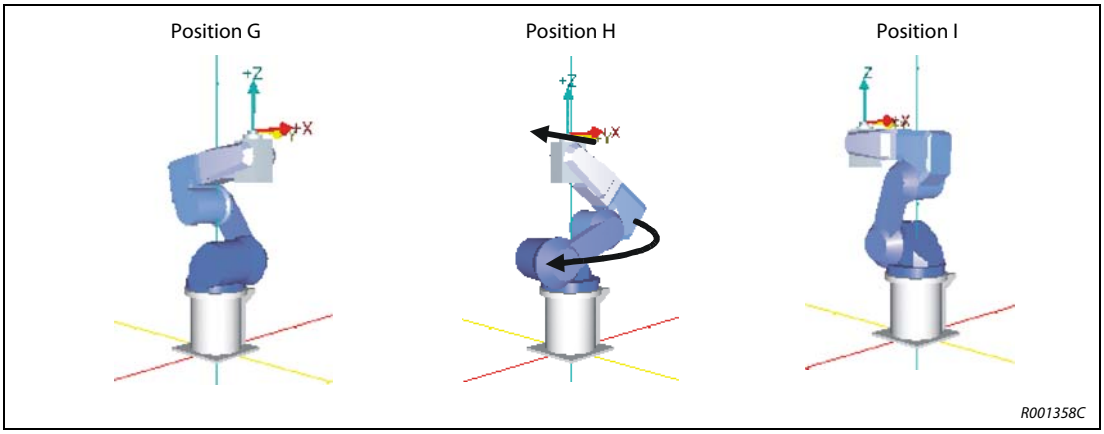


Abb. 9-46: *Verfahrweg verläuft in der Nähe eines singulären Punktes*

ACHTUNG:

Verläuft der Verfahrweg in der Nähe eines singulären Punkts, kann der Roboter in einem weiten Kreisbogen rotieren, wie in der Abbildung oben (Position H) gezeigt. Beobachten Sie daher die Roboterbewegungen genau und halten Sie sich nicht im Aktionsradius des Roboters auf wie z. B. beim Teachen von Positionen.

9.23.3 Aktivierung der Funktion zum Durchfahren singulärer Punkte

Möchten Sie die Funktion zum Durchfahren singulärer Punkte für den JOG-Betrieb aktivieren, setzen Sie Parameter FSPJOGMD auf „1". Schalten Sie anschließend die Spannungsversorgung des Steuergerätes aus und wieder ein. Aktivieren Sie die Funktion für den Automatikbetrieb, indem Sie die Konstante 2 in der TYPE-Festlegung im jeweiligen Interpolationsbefehl auf „2" setzen.

Bei der Nutzung der Funktion zum Durchfahren singulärer Punkte gelten folgende Einschränkungen:

- Die Funktion kann beim Einsatz von Zusatzachsen für einen Betrieb mehrerer Mechanismen nicht verwendet werden.
- Die Funktion kann bei einer Synchronsteuerung von Zusatzachsen nicht verwendet werden.
- Die Funktion kann bei aktivierter Achsenweichheit nicht verwendet werden.
- Die Funktion kann bei aktivierter Kollisionsüberwachung nicht verwendet werden.
- Die Datenerfassungsstufe zur Überwachung der Wartungsintervalle muss auf „1" gesetzt sein (Werkseinstellung).

9.23.4 Funktion zum Durchfahren singulärer Punkte im JOG-Betrieb

Im JOG-Betrieb wird die Funktion zum Durchfahren singulärer Punkte durch Einstellung des Parameters FSPJOGMD auf „1" aktiviert und durch Einstellung auf „0" deaktiviert.

FSPJOGMD	XYZ-JOG	TOOL-JOG	3-Achsen-XYZ-JOG	Kreis-JOG	Gelenk-JOG
0 (Werkseinstellung)	Wie vorher	Wie vorher	Wie vorher	Wie vorher	Wie vorher
1	Durchfahren singulärer Punkte im XYZ-JOG-Betrieb	Durchfahren singulärer Punkte im TOOL-JOG-Betrieb	Wie vorher	Wie vorher	Wie vorher

Tab. 9-43: *Einstellung des Parameters FSPJOGMD*

- Bei Robotern, die nicht zur Nutzung der Funktion zum Durchfahren singulärer Punkte geeignet sind, hat eine Einstellung des Parameters FSPJOGMD keine Wirkung. Der vorhergehende Betrieb wird nicht beeinflusst. (Unterstützt wird die Funktion zum Durchfahren singulärer Punkte von allen Knickarmrobotern der Serien RV-2SDB/3SDB/3SDJB/6SD/6SDL/12SD/12SDL/18SD.)
- Es ist nicht möglich mehrere Achsen gleichzeitig beim Durchfahren singulärer Punkte im JOG-Betrieb zu verfahren. Der Versuch während einer Achsenbewegung eine zweite Achse zu verfahren, wird ignoriert.
- Wird eine Achse im JOG-Betrieb mittels der Teaching Box in die Nähe eines singulären Punktes verfahren, erfolgt die Ausgabe einer Fehlermeldung (siehe auch Abschn. 9.20).
- Der Wert des Parameters FSPJOGMD ist auch bei Festlegung des JOG-Betriebs über spezielle Eingangssignale wirksam.

9.23.5 Funktion zum Durchfahren singulärer Punkte beim Anfahren definierter Positionen

Der Wert des Parameters FSPJOGMD ist auch beim Anfahren definierter Positionen wirksam.

FSPJOGMD	Mov-Bewegungsbefehl	Mvs-Bewegungsbefehl
0 (Werkseinstellung)	Wie vorher	Wie vorher
1	Wie vorher	Durchfahren singulärer Punkte

Tab. 9-44: *Durchfahren singulärer Punkte beim Anfahren definierter Positionen*

ACHTUNG:

Wird ein Interpolationsbefehl (z. B. Mvs P1) bei einer Einstellung des Parameters auf „1" (aktiv) direkt über die Teaching Box ausgeführt, ist die Funktion zum Durchfahren singulärer Punkte aktiv, auch wenn sie nicht über die TYPE-Festlegung eingestellt wurde.

9.23.6 Funktion zum Durchfahren singulärer Punkte im Automatikbetrieb

Aktivieren Sie die Funktion zum Durchfahren singulärer Punkte im Automatikbetrieb über die TYPE-Festlegung des jeweiligen Interpolationsbefehls.

9.23.7 TYPE (Type)

Funktion: Durchfahren des singulären Punktes festlegen

Definieren Sie die Durchfahrt durch den singulären Punkt in der TYPE-Festlegung des Interpolationsbefehls. Die Festlegung ist bei folgenden Interpolationsbefehlen möglich:
Linear-Interpolation (Mvs) und Kreis-Interpolation (Mvr, Mvr2 und Mvr3).

Eingabeformat

TYPE □ <Konstante 1>,<Konstante 2>

<Konstante 1> Direkte/indirekte Anfahrt = 0/1
<Konstante 2> Drehung/3-Achsen-XYZ/Durchfahren des singulären Punktes = 0/1/2

Programmbeispiel

```
1 Mvs P1 TYPE 0,2        'Position P1 mittels Linear-Interpolation und aktivierter
                         Funktion zum Durchfahren singulärer Punkte anfahren
2 Mvr P1,P2,P3 TYPE 0,2  'Position P3 mittels Kreis-Interpolation und aktivierter
                         Funktion zum Durchfahren singulärer Punkte von P1
                         aus anfahren
```

Beschreibung

- Wird die Konstante 2 bei einem Roboter, der die Funktion zum Durchfahren singulärer Punkte nicht unterstützt, auf „2" gesetzt, tritt ein Laufzeitfehler auf.
- Bei aktivierter Funktion zum Durchfahren singulärer Punkte wird der Stellungsmerker zwischen dem Start- und Endpunkt nicht geprüft. Da der Zustand des Stellungsmerkers an der Zielposition nicht bewertet wird, erfolgt auch vor der Ausführung der Bewegung keine Prüfung des Bewegungsbereiches hinsichtlich der Zielposition und der Zwischenpositionen.
- Wurde über den Befehl Spd eine Geschwindigkeit festgelegt, so entspricht diese Geschwindigkeit dem Maximalwert. In der Nähe eines singulären Punktes wird die Geschwindigkeit automatisch auf einen Wert verringert, bei dem kein Geschwindigkeitsfehler auftritt.
- Interpolationsbefehle, für die die Funktion zum Durchfahren singulärer Punkte aktiviert ist, können nicht mit optimaler Beschleunigung/Abbremsung ausgeführt werden. Die Beschleunigung/Abbremsung ist fest vorgegeben. Entspricht die Beschleunigungszeit aufgrund der Festlegung im Befehl Accel nicht der Bremszeit, wird der größere Wert sowohl für die Beschleunigung als auch für die Abbremsung verwendet.
- Eine freigegebene Cnt-Einstellung zur Steuerung kontinuierlicher Bewegungen ist bei aktivierter Funktion zum Durchfahren singulärer Punkte unwirksam. Der Roboter bewegt sich mit der festgelegten Beschleunigung/Abbremsung.
- Entspricht die aktuelle Position vor der Ausführung einer Kreis-Interpolation nicht der Startposition, erfolgt die Anfahrt des Startpunktes mittels 3-Achsen-XYZ-Linear-Interpolation, auch wenn die Funktion zum Durchfahren singulärer Punkte in der TYPE-Festlegung aktiviert wurde.
- Wird die Ausführung eines Interpolationsbefehls mit aktivierter Funktion zum Durchfahren singulärer Punkte unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter entsprechend der Einstellung im Parameter RETPATH zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Ist der Parameter RETPATH auf „0" (deaktiviert) gesetzt, erfolgt keine Rückkehr zur Position der Unterbrechung. Der Stellungsmerker wechselt seinen Zustand nicht, solange nach der Fortsetzung des Verfahrweges kein singulärer Punkt durchfahren wird (siehe folgende Abbildung). Daher kann die Stellung des Roboters nach Abschluss des Interpolationsvorgangs von der Stellung ohne Unterbrechung des Interpolationsvorgangs abweichen.

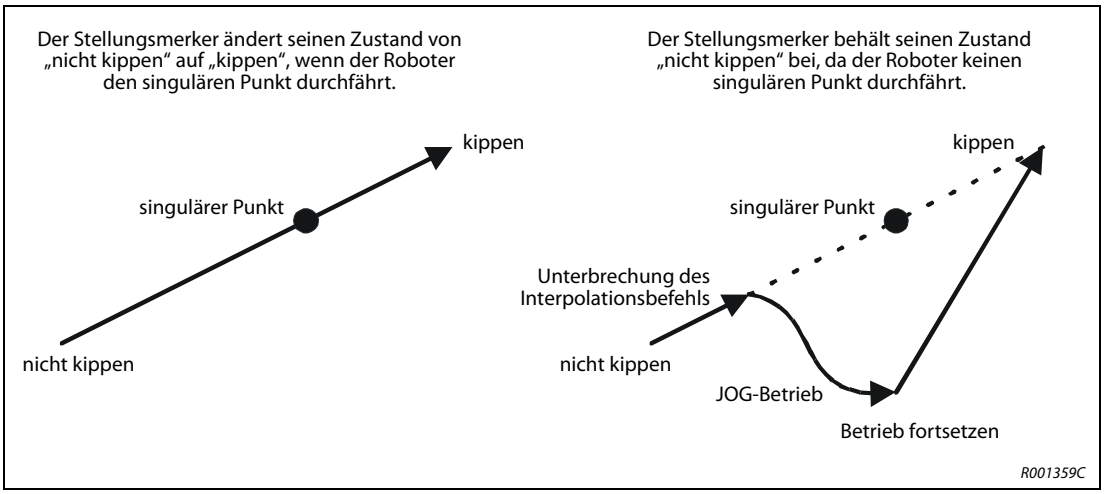

Der Stellungsmerker ändert seinen Zustand von „nicht kippen" auf „kippen", wenn der Roboter den singulären Punkt durchfährt.
kippen
singulärer Punkt
nicht kippen
Der Stellungsmerker behält seinen Zustand „nicht kippen" bei, da der Roboter keinen singulären Punkt durchfährt.
kippen
singulärer Punkt
Unterbrechung des Interpolationsbefehls
nicht kippen
JOG-Betrieb
Betrieb fortsetzen
R001359C

Abb. 9-47: *Unterbrechung des Interpolationsbefehls*

- Ist die Funktion zum Durchfahren singulärer Punkte aktiviert, kann die Geschwindigkeit geringer sein, als bei einer normalen Interpolation. Da die Funktion komplizierte Berechnungsprozesse erfordert, kann die gesamte Programmverarbeitungszeit ansteigen. Verwenden Sie daher die Funktion zum Durchfahren singulärer Punkte nur dann, wenn sie wirklich nötig ist.

9.23.8 Kollisionsüberwachung

9.23.9 Funktionsbeschreibung

Während des Roboterbetriebs kann es durch Fehler des Bedieners, Fehler im Ablaufprogramm usw. zu Kollisionen mit Werkstücken oder anderen Vorrichtungen kommen. Normalerweise stoppt der Roboter in solchen Fällen über die Schutzfunktionen der Servoantriebe für die Robotermotoren (wie z. B. die Positionsabweichungserkennung), um Beschädigungen der Roboterhand und des -arms sowie des Werkstücks und anderer Vorrichtungen zu vermeiden. Da aber Roboter mit immer höheren Geschwindigkeiten und stärkeren Motoren arbeiten, wird es immer schwieriger, Kollisionen allein nur durch die Überlastung der Servoantriebe zu erkennen. Die nachfolgend beschriebene Kollisionsüberwachung reagiert hier wesentlich sensibler als die normale Schutzfunktion des Servoantriebs und stoppt den Roboter zur Vermeidung von Schäden viel schneller.

GEFAHR:

Auch wenn die Kollisionsüberwachung aktiviert ist, kann nicht verhindert werden, dass der Bediener beim Zusammenstoß mit dem sich bewegenden Roboterarm verletzt wird. Die gültigen Sicherheits- und Unfallverhütungsvorschriften müssen in jedem Fall eingehalten werden, egal ob die Kollisionsüberwachung aktiviert ist oder nicht.

ACHTUNG:

Auch wenn die Kollisionsüberwachung aktiviert ist, kann nicht verhindert werden, dass Schäden am Roboter, an der Roboterhand und am Werkstücks durch einen Zusammenstoß mit umgebenden Vorrichtungen auftreten. Seien Sie daher immer besonders vorsichtig, wenn Sie den Roboter in Betrieb nehmen oder daran arbeiten, um Zusammenstöße mit anderen Gegenständen zu vermeiden.

Arbeitsweise der Kollisionsüberwachung

Wenn der Roboter mit einem Gegenstand zusammenstößt, kommt es zu einer Positionsabweichung der einzelnen Gelenke und der Roboter kann den Positionsvorgaben nicht mehr folgen. Weiterhin wird von der Servosteuerung ein höheres Drehmoment zurückgemeldet. Solange die Kollisionsursache nicht beseitigt ist, steigt das erzeugte Drehmoment weiter an und wird dadurch größer, als im Normalfall erwartet wird.

Für die Kollisionsüberwachung macht man sich diese Eigenschaft des Servoantriebs zunutze. Zuerst wird das benötigte Drehmoment als Sollwert für den Betrieb unter normalen Lastbedingungen für jede Achse festgelegt. Danach wird während des Betrieb für jede Achse das aktuell benötigte Drehmoment (Istwert) ständig mit dem zuvor festgelegten Drehmoment verglichen. Überschreitet die Drehmomentabweichung nun einen bestimmte Grenzwert (Schwellwert), wird angenommen, dass eine Kollision aufgetreten ist. Der Servoantrieb schaltet jetzt unverzüglich ab und der Roboter stoppt.

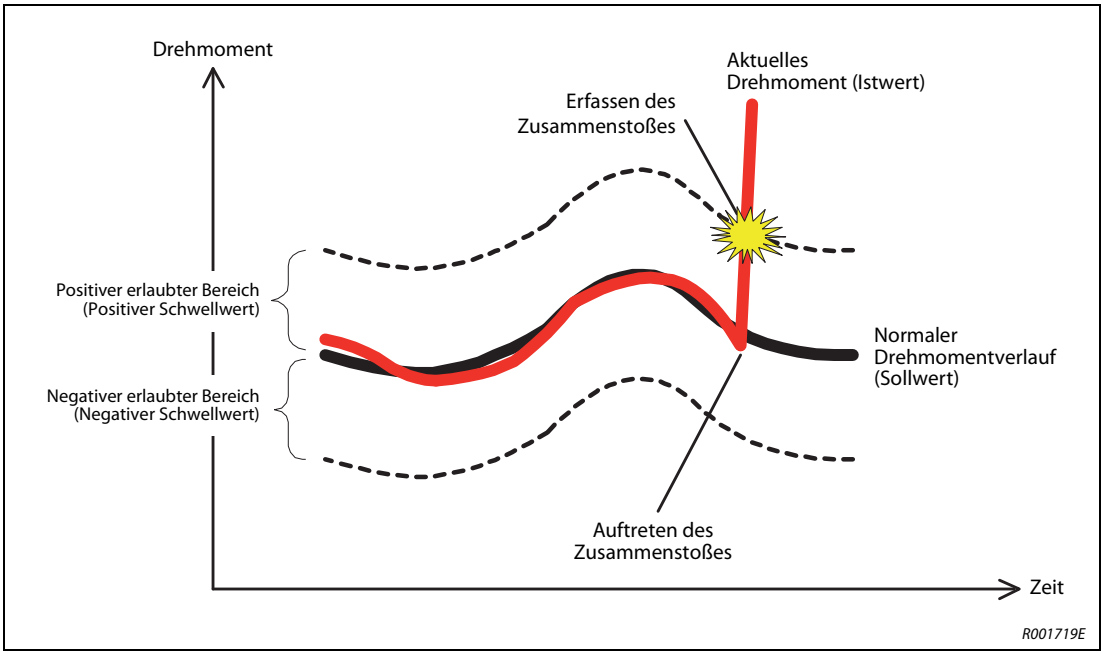


Abb. 9-48: *Prinzip der Kollisionsüberwachung*

9.23.10 Robotermodelle mit Kollisionsüberwachung

Die Kollisionsüberwachung steht bei den folgenden Robotermodellen zur Verfügung:

- RV-2SDB/3SDB/3SDJB/6SD/6SDL/12SD/12SDL
- RH-6SDH/12SDH/20SDH
- RH-3SDHR
- RV-2SQB/3SQB/3SQJB/6SQ/6SQL/12SQ/12SQL
- RH-6SQH/12SQH/20SQH
- RH-3SQHR

9.23.11 Parameter in Zusammenhang mit der Kollisionsüberwachung

Die folgenden Parameter sind mit der Kollisionsüberwachung verknüpft. Eine detaillierte Beschreibung dieser Parameter finden Sie in Abschn. 9.2 „Bewegungsparameter“ und in Abschn. 9.19 „Hand- und Werkstückbedingungen“.

Parameter	Beschreibung	Werkseinstellung
COL	Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung 1. Element: Kollisionsüberwachung aktivieren (1) oder deaktivieren (0) 2. Element: Status direkt nach Einschalten der Spannungsversorgung aktiviert (1)/deaktiviert (0) 3. Element: Freigabe im JOG-Betrieb aktiviert (1)/deaktiviert (0)/ NOERR-Modus (2)	Für die Robotermodelle RV-SD/SQ: 0,0,1 RH-SDH/SQH: 1,0,1 RH-SDHR/SQHR: 1,1,1
COLLVL	Einstellung der Ansprechschwelle im Programmbetrieb für jede Achse. Die Ansprechschwelle ist ein abgestufter Wert als Vergrößerungsfaktor der Standardansprechschwelle. Je kleiner der Wert, desto höher ist die Ansprechschwelle Einstellbereich: 1 bis 500 %, Einheit: %	Einstellung ist abhängig vom Robotermodell
COLLVLJG	Einstellung der Ansprechschwelle (Empfindlichkeit) im JOG-Betrieb (inklusive Unterbrechung) für jede Achse. Die Ansprechschwelle ist ein abgestufter Wert als Vergrößerungsfaktor der Standardansprechschwelle. Je kleiner der Wert, desto höher ist die Ansprechschwelle. Einstellbereich: 1 bis 500 %, Einheit: %	Einstellung ist abhängig vom Robotermodell
HNDDAT1 : HNDDAT8	Einstellung der Handbedingungen (Festlegung im Werkzeugkoordinatensystem) Mit HNDDAT0 werden die hier festgelegten Bedingungen nach Einschalten der Spannungsversorgung verwendet. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Einstellung ist abhängig vom Robotermodell
WRKDAT1 : WRKDAT8	Einstellung der Werkstückbedingungen (Festlegung im Werkzeugkoordinatensystem) Mit WRKDAT0 werden die hier festgelegten Bedingungen nach Einschalten der Spannungsversorgung verwendet (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
HNDHOLD1 : HNDHOLD8	Festlegung, ob die Hand beim Befehl HOpen und HClose geschlossen (1) oder nicht geschlossen (0) wird. 1. Element: Einstellung des Status für den Befehl HOpen 2. Element: Einstellung des Status für den Befehl HClose	0,1

Tab. 9-45: *Mit der Kollisionsüberwachung verknüpfte Parameter*

9.23.12 Einsatz der Kollisionsüberwachung

Um die Kollisionsüberwachung verwenden zu können, muss zuerst das 1. Element des Parameters COL auf „1“ (aktivieren) eingestellt werden. Schalten Sie Spannungsversorgung der Robotersteuerung aus und wieder ein. Stellen Sie als nächstes für den JOG- und den Automatikbetrieb die Parameter der Kollisionsüberwachung ein (Aktivierung/Deaktivierung der Funktion und Einstellung des Schwellwerts). Beachten Sie dabei Tab. 9-45, wo die entsprechenden Parameter für die Kollisionsüberwachung aufgeführt sind.

Einsatz der Kollisionsüberwachung im JOG-Betrieb

Alle Einstellungen der Kollisionsüberwachung während des JOG-Betriebs erfolgen mit Parametern. Dabei werden Einstellungen von Parametern erst dann wirksam, wenn die Steuerung nach der Änderung aus- und wieder eingeschaltet wird. Die folgende Tabelle zeigt die Parameter, die für eine Kollisionsüberwachung im JOG-Betrieb in Frage kommen.

Parameter	Beschreibung	Werkseinstellung
COL	Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung 1. Element: Kollisionsüberwachung aktivieren (1) oder deaktivieren (0) 3. Element: Freigabe im JOG-Betrieb aktiviert (1)/deaktiviert (0)/ NOERR-Modus (2)	Für die Robotermodelle RV-SD/SQ: 0,0,1 RH-SDH/SQH: 1,0,1 RH-SDHR/SQHR: 1,1,1
COLLVLJG	Einstellung der Ansprechschwelle (Empfindlichkeit) im JOG-Betrieb (inklusive Unterbrechung) für jede Achse. Die Werkseinstellung für das Robotermodell RV-3SDB ist beispielsweise: 100, 100, 100, 100, 100, 100, 100, 100	Einstellung ist abhängig vom Robotermodell
HNDDAT0	Einstellung der Handbedingungen Die Werkseinstellung für das Robotermodell RV-3SDB ist beispielsweise: 3.5,284.0,284.0,286.0,0.0,0.0,75.0	Einstellung ist abhängig vom Robotermodell
WRKDAT0	Einstellung der Werkstückbedingungen (Festlegung im Werkzeugkoordinatensystem)	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

Tab. 9-46: *Parameter der Kollisionsüberwachung bei JOG-Betrieb*

Einstellung der Ansprechschwelle

Standardmäßig ist die Ansprechschwelle der Kollisionsüberwachung relativ niedrig eingestellt. Zur Erhöhung der Empfindlichkeit ist die Ansprechschwelle über den Parameter COLLVLJG zu erhöhen. Achten Sie auch auf eine korrekte Einstellung der Parameter HNDDAT0 und WRKDAT0, um ein ungewolltes Ansprechen der Kollisionsüberwachung zu vermeiden.

HINWEIS Wenn die Empfindlichkeit zu hoch eingestellt ist (kleiner Einstellwert), kann die Kollisionsüberwachung in Abhängigkeit der Roboterstellung ungewollt ansprechen. Stellen Sie in diesem Fall die Empfindlichkeit niedriger ein (Vergrößerung des Einstellwerts).

Reaktion auf eine erkannte Kollision

Erfasst das Steuergerät einen Zusammenstoß des Roboters mit einer umliegenden Einrichtung, erfolgt die Ausgabe der Fehlernummer 101n (die letzte Stelle „n“ gibt die Achsennummer wieder) und der Roboter stoppt, sobald der Servo abschaltet. Wurde im 3. Element des Parameters COL der Wert „2“ (NOERR) eingestellt, wird kein Fehler ausgegeben und der Roboter stoppt nach Abschalten der Servos (hier wird der entsprechende Fehler nur in der Fehlerliste abgespeichert).

Kollisionsüberwachung während des JOG-Betriebs vorübergehend deaktivieren

Halten Sie die [Reset]-Taste der Teaching Box betätigt, während Sie die Servoversorgungsspannung einschalten und den JOG-Betrieb ausführen. Solange Sie die [Reset]-Taste betätigen, ist die Kollisionsüberwachung deaktiviert.

Betrieb nach einem Zusammenstoß

Wird der Servoantrieb nach der erkannten Kollision wieder eingeschaltet, während der Roboterarm oder die -hand den Gegenstand weiterhin berührt, mit dem der Zusammenstoß zuvor statt fand, spricht die Kollisionsüberwachung erneut an. Erscheint bei mehreren Einschaltversuchen des Servoantriebs ein Fehler, bewegen Sie den Arm durch Lösen der Bremse oder per JOG-Betrieb vom Ort der Kollison weg, damit der Roboter nicht mehr in seiner Bewegung behindert wird (siehe auch Abschn. 3.13 „Fehler temporär zurücksetzen“).

Einsatz der Kollisionsüberwachung bei Automatikbetrieb

Der Status der Kollisionsüberwachung bei Automatikbetrieb direkt nach Einschalten der Spannungsversorgung wird mit einem Parameter festgelegt. In der Praxis wird die entsprechende Funktion in einem Programm mit Befehlen aus MELFA-BASIC V eingestellt. Die folgende Tabelle zeigt die Einstellung des Status direkt nach Einschalten der Spannungsversorgung und die Parameter der Kollisionsüberwachung bei Automatikbetrieb. Weitere Angaben zu den MELFA-BASIC-V-Befehlen finden Sie in Kap. 6 und zu den Roboterstatusvariablen in Abschn. 7.2.

Parameter	Beschreibung	Werkseinstellung
COL	Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung 1. Element: Kollisionsüberwachung aktivieren (1) oder deaktivieren (0) 2. Element: Status direkt nach Einschalten der Spannungsversorgung bei Automatikbetrieb, aktiviert (1)/deaktiviert (0)	Für die Robotermodelle RV-SD/SQ: 0,0,1 RH-SDH/SQH: 1,0,1 RH-SDHR/SQHR: 1,1,1
COLLVL	Einstellung der Ansprechschwelle (Empfindlichkeit) im JOG-Betrieb (inklusive Unterbrechung) für jede Achse. Die Werkseinstellung für das Robotermodell RV-3SDB ist beispielsweise: 100, 100, 100, 100, 100, 100, 100, 100	Einstellung ist abhängig vom Robotermodell
HNDDAT1 : HNDDAT8	Einstellung der Handbedingungen Die Werkseinstellung für das Robotermodell RV-3SDB ist beispielsweise: 3.5,284.0,284.0,286.0,0.0,0.0,75.0	Einstellung ist abhängig vom Robotermodell
WRKDAT1 : WRKDAT8	Einstellung der Werkstückbedingungen (Festlegung im Werkzeugkoordinatensystem)	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
HNDHOLD1 : HNDHOLD8	Festlegung, ob die Hand beim Befehl HOpen und HClose geschlossen (1) oder nicht geschlossen (0) wird.	0,1

Tab. 9-47: *Mit der Kollisionsüberwachung verknüpfte Parameter bei Automatikbetrieb*

Befehl/ Parameter	Beschreibung
ColChk	Der Befehl aktiviert die Kollisionsüberwachung und legt den NOERR-Modus fest.
COLLVL	Einstellung der Ansprechschwelle im Programmbetrieb für jede Achse. Die Ansprechschwelle ist ein abgestufter Wert als Vergrößerungsfaktor der Standardansprechschwelle. Einheit: %. Beispiel: COLLVL 80, 120, 120, 120, 50, 80 'Festlegung der Ansprechschwelle für die Achsen J1 bis J6
LoadSet	Legt die Hand- und Werkstückbedingungen fest. Diese Einstellung ist notwendig, wenn die verwendete Hand oder das zu greifende Werkstück während des Automatikbetriebs geändert werden muss. Beispiel: LoadSet 1, 0 'Festlegung der Bedingungen der Parameter HNDDAT1 und WRKDAT0
J_ColMxl	Die Variable enthält die auf die Ansprechschwelle umgerechnete maximale Drehmomentabweichung zwischen dem Drehmoment-Istwert und dem Drehmoment-Sollwert bei aktivierter Kollisionsüberwachung. Die Drehmomentabweichung wird auf die einzelnen Achseneinstellungen des Befehls COLLVL bezogen.
M_ColSts	Die Variable meldet den Status der Kollisionsüberwachung. Der Wert bei einer Kollision ist „1". Im NOERR-Modus wird die Variable als Interrupt-Bedingung verwendet.
P_ColDir	Die Variable enthält die Richtung des Verfahrweges bei Ansprechen der Kollisionsüberwachung (als Verhältniswerte in X-, Y- und Z-Richtung). Sie wird für den Rückzug im NOERR-Modus verwendet.

Tab. 9-48: *Mit der Kollisionsüberwachung verknüpfte MELFA-BASIC-V-Befehle und Parameter bei Automatikbetrieb*

HINWEISE

Wenn die Kollisionsüberwachung über das gesamte Programm hinweg aktiviert ist, ist die Wahrscheinlichkeit sehr hoch, dass irrtümlich Kollisionen erkannt werden, wo keine vorhanden sind. In diesem Fall müsste die Ansprechschwelle für alle Abläufe verringert werden. Das hätte aber zur Folge, dass auch die Ansprechschwelle für solche Abläufe verringert wird, für die eine Kollisionserkennung wirklich wichtig ist. Eine gute Lösung könnte hier sein, dass die Kollisionsüberwachung nur für die wichtigen Abläufe aktiviert wird, wodurch die hohe Empfindlichkeit der Kollisionsüberwachung beibehalten werden kann.

In Abhängigkeit vom Programm kann sich die Zykluszeit verlängern, wenn die Kollisionsüberwachung aktiviert ist. Um dies zu vermeiden, sollte die Kollisionsüberwachung nur für solche Abläufe eingesetzt werden, bei denen tatsächlich Zusammenstöße auftreten können, anstatt diese im gesamten Programm zu verwenden.

Einstellung der Ansprechschwelle

Die Ansprechschwelle (Empfindlichkeit) der Kollisionsüberwachung wird im Automatikbetrieb entsprechend dem Betriebsablauf des Roboters eingestellt. Nachfolgend wird ein Einstellbeispiel der Ansprechschwelle gezeigt. Achten Sie auch auf eine korrekte Einstellung der Hand- und Werkstückbedingungen, um ein ungewolltes Ansprechen der Kollisionsüberwachung zu vermeiden.

Reaktion auf eine erkannte Kollision

Wird während des Automatikbetriebs ein Zusammenstoß des Roboters mit einer umliegenden Einrichtung erfasst, erfolgt die Ausgabe der Fehlernummer 101n (die letzte Stelle „n" gibt die Achsennummer wieder) und der Roboter stoppt, sobald die Servos abschalten. Wurde im 3. Element des Parameters COL der Wert „2" (NOERR) eingestellt, wird kein Fehler ausgegeben und der Roboter stoppt nach Abschalten der Servos (hier wird der entsprechende Fehler nur in der Fehlerliste abgespeichert).

Beispiel ▽ Einstellablauf der Ansprechschwelle der Kollisionsüberwachung beim Automatikbetrieb

① Fügen Sie vor und nach den Abläufen, die auf Kollision überwacht werden sollen, den Parameter COLLVL und den Befehl ColChk ein.

② Stellen Sie die Empfindlichkeit niedrig ein (ein hoher Wert, wie z. B. 300 für die Achsen im Parameter COLLVL), um ein irrtümliches Ansprechen zu verhindern.

③ Starten Sie das Programm und beobachten Sie über J_ColMxl die maximale Drehmomentabweichung vom Sollwert. Beachten Sie, dass der Wert schwanken kann. Wiederholen Sie den Normalbetrieb mehrmals und notieren Sie jedesmal den Wert der Variable J_ColMxl.

④ Suchen Sie aus allen aufgezeichneten Werten der Variable J_ColMxl für jede Achse den Maximalwert heraus, und addieren Sie auf diese Werte eine Aufschlag von z. B. 20 %.

⑤ Stellen Sie die in Schritt ④ ermittelten Werte für alle Achsen im Parameter COLLVL ein und lassen Sie das Programm erneut ablaufen. Kontrollieren Sie, dass für die Abläufe mit Kollisionsüberwachung im Normalbetrieb keine irrtümlichen Zusammenstöße erkannt werden. Werden diese trotzdem erkannt, muss der Wert der Variable J_ColMxl zur Herabsetzung der Empfindlichkeit schrittweise erhöht werden.

⑥ Wiederholen Sie Schritt ⑤ so oft, bis im Normalbetrieb keine irrtümlichen Kollisionen mehr erkannt werden.

△

HINWEISE

Bei Änderungen der Ablaufgeschwindigkeit kann es sein, dass die Ansprechschwellen der Kollisionsüberwachung angepasst werden müssen. Betreiben Sie den Roboter mit der neuen Geschwindigkeit und passen Sie die Ansprechschwellen an.

Bei Einsatz der Kollisionsüberwachung für mehrere Roboter kann es sein, dass die Ansprechschwellen bei jedem Roboter individuell für den gleichen Ablauf eingestellt werden müssen. Das liegt daran, dass sich die Roboter durch verschiedene Motorkenndaten und Einsatzumgebungen unterscheiden können.

Programmbeispiel

Mit diesem Programm wird der Roboter nach einem Zusammenstoß über Interrupt-Verarbeitung zu einem Rückzugspunkt gefahren.

```
1  Def Act 1,M_ColSts(1)=1 GoTo *HOME,S   'Festlegung des Ablaufs nach erkannter
                                          Unterbrechung durch Zusammenstoß.
2  Act 1=1
3  ColLvl 80,120,120,100,80,80,,          'Einstellung der Ansprechschwellen
4  ColChk ON,NOERR                        'Aktivierung der Kollisionsüberwachung im
                                          Modus NOERR
5  Mov P1
6  Mov P2                                 'Sprung zur Interrupt-Verarbeitung bei Erkennen
                                          einer Kollison während Abarbeitung der Schritte 5 bis 8
7  Mov P3
8  Mov P4
9  ColChk OFF                             'Deaktivierung der Kollisionsüberwachung
10 Act 1=0
   :
   :
1000 *HOME                                'Interrupt-Verarbeitung bei Kollisionserkennung
1001 ColChk OFF                           'Deaktivierung der Kollisionsüberwachung
1002 Servo On                             'Servo einschalten
1003 PESC=P_ColDir(1)*(-5)                'Berechnung der Rückzugstrecke von ca. 5 mm
1004 PDst=P_Fbc(1)+PESC                   'Erzeugung des Rückzugspunkts
1005 Mvs PDst                             'Zum Rückzugspunkt fahren
1006 Error 9100                           'Unterbrechung des Betriebs und Ausgabe eines
                                          benutzerdefinierten Fehlers mit niedrigem Fehler-
                                          status (L-Level)
```

HINWEIS

Die Kollisionsüberwachung erkennt Zusammenstöße, indem Sie die üblichen Drehmomente, die bei Ausführung eines Bewegungsbefehls an einer Achse auftreten, unter Berücksichtigung der Last u.Ä. mit den tatsächlich auftretenden Drehmomenten vergleicht.
Wirken auf den Roboter im Normalbetrieb externe Kräfte ein, kann es deshalb vorkommen, dass die Kollisionsüberwachung anspricht, obwohl kein Zusammenstoß aufgetreten ist. Dies kann z.B. der Fall sein, wenn sich die Roboterhand in den Pneumatikschläuchen oder der Verkabelung verfängt. Je nach Größe der auftretenden Kräfte spricht dann die Kollisionsüberwachung an.
Stellen Sie daher sicher, dass ein Ansprechen der Kollisionsüberwachung nicht durch extern einwirkende Kräfte hervorgerufen wurde, sondern tatsächlich Folge eines Zusammenstoßes ist.

Unterschied der Kollisionsüberwachung bei JOG- und Automatikbetrieb

Die Geschwindigkeit der Abläufe des Roboters unterscheidet sich zwischen JOG- und Automatikbetrieb. Diese beiden Betriebsarten wurden bewusst getrennt, um die Kollisionserkennung für jeden der beiden Fälle zu optimieren.

Die Bedeutung der beiden Betriebsarten ist wie folgt festgelegt:

– bei JOG-Betrieb: bedeutet bei JOG-Betrieb oder bei Unterbrechung des Automatikbetriebs

– bei Automatikbetrieb: bedeutet bei Automatikbetrieb, bei schrittweiser Ausführung vorwärts/rückwärts oder bei Prüfung der Positionsdaten

Den jeweiligen Status bei Ausführen der vorgenannten Betriebsarten zeigt die folgende Abbildung.

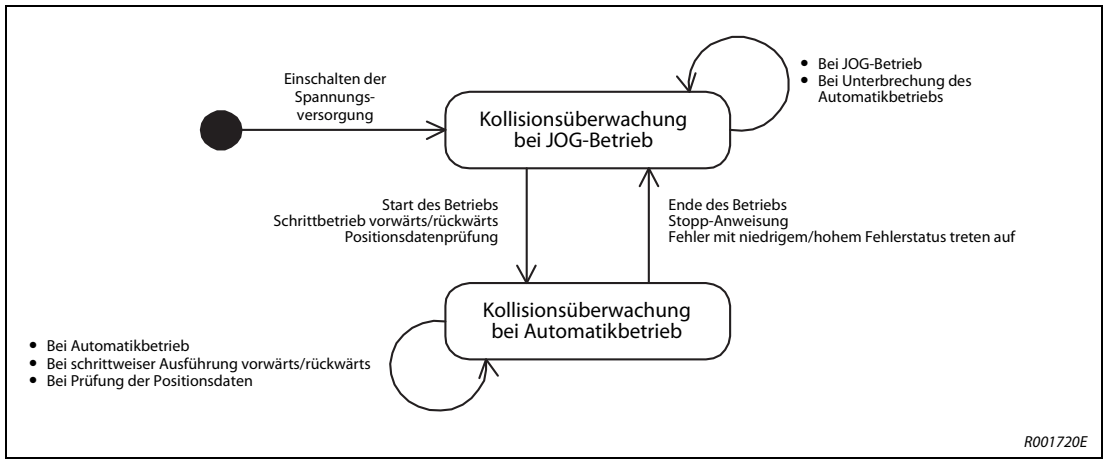


Abb. 9-49: *Statusumschaltung zwischen Automatik- und JOG-Betrieb*

Wird Automatikbetrieb mit deaktivierter Kollisionsüberwachung zur Unterbrechung die Stopp-Taste betätigt, wenn für den JOG-Betrieb die Kollisionsüberwachung aktiviert ist, dann ist während der Unterbrechung des Automatikbetriebs trotzdem die Kollisionsüberwachung aktiviert.

Kollisionsüberwachung bei abgeschaltetem Servo

Sowohl beim Automatikbetrieb, als auch beim Automatikbetrieb ist die Kollisionsüberwachung zeitweise deaktiviert, während der Servo abgeschaltet ist.

10 Externe Ein-/Ausgänge

10.1 Einteilung

Die externen Ein-/Ausgänge sind in 3 Gruppen aufgeteilt:

- Spezielle Ein-/Ausgänge
  Die Ein-/Ausgänge dienen zur Steuerung und Statusanzeige des Roboters. Häufig verwendete Funktionen sind dabei vordefiniert. Der Anwender hat die Möglichkeit, neue Funktionen hinzuzufügen und bestehende Funktionen zu modifizieren.
- Allgemeine Ein-/Ausgänge
  Die Ein-/Ausgänge dienen zur Kommunikation mit Peripheriegeräten über das Roboterprogramm und können frei programmiert werden. Sie ermöglichen die Überwachung der Roboterposition und der Übertragung von Positioniersignalen von externen Geräten (z. B. SPS).
- Ein-/Ausgänge für Greifhand
  Die Ein-/Ausgänge können zur Unterstützung von Handfunktionen programmiert werden. Sie dienen z. B. zum Öffnen und Schließen der Hand oder zum Einlesen von Handsensorsignalen. Dazu benötigen Sie das optionale Steuermodul für die pneumatische/elektrische Greifhand. Die Steuerung und Überwachung der Eingänge erfolgt über das Roboterprogramm.

ACHTUNG:

Sie können die Spezial-Eingänge während der Programmausführung in allgemeine Eingänge umdefinieren. Das ist aus Sicherheitsgründen nur für die numerischen Dateneingänge zu empfehlen. Dagegen können Sie die Spezialausgänge nicht als allgemeine Ausgänge im Programm benutzen. Bei einem Versuch löst der Roboter Alarm aus.

10.1.1 Allgemeine Übersicht der Ein- und Ausgänge

	E/A-Signalnummer	Zugriff
E/A für die Greifhand	900–907	Über die Variablen M_In, M_Inb, M_Inw, M_Out, M_Outb, M_Outw Auch mit den Befehlen HOpen, HClose Beispiel: If M_In(900) Then M_Out(900) = 1
E/A-Link für die SPS	10000–18191	Über die Variablen M_In, M_Inb, M_Inw, M_Out, M_Outb, M_Outw Beispiel: If M_In(10080)=1 Then M_Out(10080) = 1 HINWEIS: Eine Ausgabe mit den Variablen M_Out, M_Outb oder M_Outw ist nicht für die Signale möglich, denen spezielle Ausgänge zugeordnet wurden

Tab. 10-1: *Übersicht der Ein- und Ausgänge*

10.2 E/A-Link-Funktion für die SPS

Diese Funktion ist nur für die Steuergeräte der Modellreihe CRnQ-700 verfügbar. Die QnUD(H)CPU (diese CPU wird im weiteren Verlauf mit SPS-CPU bezeichnet) und die Q172DRCPU (diese CPU wird im weiteren Verlauf mit Roboter-CPU bezeichnet) greifen auf gemeinsamen Speicher für beide CPUs zu und Kommunikation geschieht über einen Kontaktplan. Der gemeinsame High-Speed-Speicherbereich für Multi-CPUs wird für die Kommunikation verwendet. Die Roboter-CPU nutzt die Signalnummern 10000 bis 18191 als Ein- und Ausgangssignale.

Details zu Multi-CPUs und zum gemeinsamem High-Speed-Speicherbereich finden Sie in der Bedienungsanleitung der QCPU (Multi-CPU Edition)

10.2.1 Parametereinstellung

Zur Nutzung der SPS-Link-Funktion müssen sowohl für die Roboter-, als auch für die SPS-CPU Multi-CPU-Parameter eingestellt werden.

Bei der Roboter-CPU können die Parameter über die Software RT ToolBox2 oder die Teaching-Box (R32TB, R56TB) eingestellt werden. Die Einstellung der Parameter für die SPS-CPU erfolgt mit der Software GX-Developer. Weitere Einzelheiten dazu finden Sie in den zugehörigen Bedienungsanleitungen.

Parametereinstellung der SPS

Verwenden Sie zur Einstellung der Multi-CPU-Parameter die Einstellsoftware GX-Developer.

- CPU-Anzahl
  Stellen Sie die Anzahl der CPUs ein, die in dem Standard-Baugruppenträger des Multi-CPU-Systems installiert sind.
- Synchroner Multi-CPU-Start
  Roboter-CPUs benötigen nach dem Einschalten der Spannungsversorgung einige Zeit zum Aufstarten, so dass hier die Klickbox für den synchronen Start beim Multi-CPU-System aktiviert werden sollte.
- Bereich für die High-Speed-Kommunikation für Multi-CPUs
  Einstellung der Adressanzahl in der Einheit kWorte. Die Roboter-CPU benötigt maximal einen Adressbereich von 1 kWorte, so dass hier dieser Wert eingestellt werden sollte.

Für die High-Speed-Kommunikation kann auch ein anwenderfreier Bereich und ein Bereich mit automatischer Aktualisierung eingestellt werden. Da die Roboter-CPU (Q172DRCPU) keine automatische Aktualisierung unterstützt muss die Anzahl der Adressen für diesen Bereich immer auf „0“ eingestellt werden. Beachten Sie auch zu den jeweiligen Einstellungen jeder verwendeten CPU die zugehörigen Bedienungsanleitungen.

Parametereinstellung der Roboter-CPU

Verwenden Sie zur Einstellung der Multi-CPU-Parameter die Software RT ToolBox2.

<table>
<tr><th>Parameter</th><th>Beschreibung</th><th>Werkseinstellung</th></tr>
<tr><td>QMLTCPUN</td><td>Anzahl der Multi-CPUs
Bei Multi-CPU-Systemen muss die Anzahl der CPUs eingestellt werden, die auf dem Standard-Baugruppenträger installiert sind
Einstellbereich: 1–4</td><td>2</td></tr>
<tr><td>QMLTCPU1
:
QMLTCPU4</td><td>Adressbereich der Muli-CPU bei Highspeed-Kommunikation
Bei Multi-CPU-Systemen muss die Anzahl der Adressen eingestellt werden, mit denen die Datenübertragung zwischen den CPUs Nr. 1 bis 4 über die High-speed-Kommunikation statt findet.
Es ist darauf zu achten, dass für alle CPUs die Parameter übereinstimmend eingestellt werden. Andernfalls gibt die SPS einen Fehler aus. Überprüfen Sie die Einstellungen daraufhin.
Element 1: Größe des freien Anwenderadressbereichs (1 k = 1024 Adressen)
Einstellbereich: 1 k–14 k
(Der Maximalwert hängt von der Anzahl der eingesetzen CPUs ab; siehe Tabelle)
<table><tr><th>Anzahl CPUs</th><th>Einstellbereich in Adressen</th></tr><tr><td>2</td><td>0–14 k</td></tr><tr><td>3</td><td>0–13 k</td></tr><tr><td>4</td><td>0–12 k</td></tr></table>Element 2: Anzahl Adressen für automatische Aktualisierung
Einstellbereich: 0–14335
Die Roboter-CPU unterstützt keine automatische Aktualisierung, so dass die Adressanzahl hier auf „0“ eingestellt bleibt.
Element 3: Größe des Systembereichs
Einstellbereich: 1 oder 2 (1024 oder 2048 Adressen)
Element 4: Synchroner Multi-CPU-Start
Einstellbereich: 1: Ja oder 2: Nein
Roboter-CPUs benötigen einige Zeit zum Aufstarten, so dass die Werkseinstellung von „1“ (Synchroner CPU-Start) hier nicht geändert werden sollte.</td><td>1,0,1,1</td></tr>
<tr><td>IQMEM ①</td><td>Erweiterte Funktionen bei gemeinsamer Nutzung des Speichers
Die Funktion wird bitweise zugewiesen.
1/0: aktiviert/deaktiviert
15 0
00000000 00000000
I I..... Bit 0: Erweiterte Funktionen bei gemeinsamer
I Nutzung des Speichers aktiviert
I..... Bit 1: direkte Ausführung des Ablaufprogramms
I..... Bit 2–15: keine Verwendung</td><td>0000000000000000</td></tr>
<tr><td>IQSPEC ①</td><td>Setup der Funktionen für die SQ-Serie
Die Funktion wird bitweise zugewiesen.
1/0: aktiviert/deaktiviert
15 0
00000000 00000000
I..... Bit 0: Richtung zum Beschreiben des gemeinsamen Speichers
= 0: Das Auslesen und Schreiben beginnt mit der Kopfadresse und endet mit der letzten Adresse.
= 1: Das Auslesen beginnt mit der Kopfadresse und endet mit der letzten Adresse.
Das Schreiben beginnt mit der letzten Adresse und endet mit der Kopfadresse.
I..... Bit 2–15: keine Verwendung</td><td>0000000000000001</td></tr>
</table>

Tab. 10-2: *Parametereinstellung der Roboter-CPU*

① Die Funktion steht bei den Robotern der SQ-Serie ab Software-Version N8 zur Verfügung.

Einsetzbare Multi-CPUs

Die nachfolgenden CPUs und Baugruppenträger sind zur iQ-Plattform kompatibel (Stand August 2010):

Funktion der CPU	Modell	Bemerkungen
SPS-CPU	Universal SPS-CPU Q03UD (E) CPU, Q04UD (E) HCPU, Q06UD (E) HCPU, Q10UD (E) HCPU, Q13UD (E) HCPU, Q20UD (E) HCPU, Q26UD (E) HCPU, Q50UDEHCPU, Q100UDEHCPU	• Der Baugruppenträger muss über einen Hochgeschwindigkeitsbus für den Multi-CPU-Betrieb verfügen • Die erste CPU muss die SPS-CPU sein
Roboter-CPU	Q172DRCPU	
Motion-CPU	Q172DCPU / Q173DCPU	
NC-CPU	Q172NCCPU	
Baugruppenträger	Hauptbaugruppenträger für Hochgeschwindigkeits-Kommunikation zwischen den CPUs Q38DB, Q312DB	Der Baugruppenträger muss über einen Hochgeschwindigkeitsbus für den Multi-CPU-Betrieb verfügen

Tab. 10-3: *Kompatible CPUs für Multi-CPUs der iQ-Plattform*

Die Parametereinstellung der Roboter-CPU erfolgt mit der Software RT ToolBox2 oder der Teaching-Box (R32TB, R56TB, der SPS-CPU mit der Software GX-Developer, der Motion-CPU mit der Software MT-Developer, der NC-CPU mit der dezentralen Überwachungssoftware (Remote Monitor Tool) usw. Beachten Sie auch die Bedienungsanleitungen der jeweiligen Software.

10.2.2 Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale

Bei Multi-CPU-Systemen wird für die E/A-Signale auf gemeinsamen Speicher der SPS zugegriffen, wie z. B. auf U3E0\G10000. Die Roboter-CPU mit der Nr. n greift beispielsweise auf den gemeinsamen Speicher U3En\G1000 zu. (n = 1 bis 3; es können maximal 3 Roboter-CPUs eingesetzt werden.)

Die E/A-Signalnummer der drei Roboter-CPUs sind alle von 10000 bis 18191. Seitens der SPS werden Wort-Operanden verwendet, seitens der Roboter-CPU Bit-Operanden, so dass hier erhöhte Vorsicht geboten ist.

Die folgende Tabelle zeigt den gemeinsamen CPU-Speicher und die Kompatibilität der Roboter-E/A-Signale. Beachten Sie, dass die Zuordnung nicht geändert werden kann.

SPS (Wort-Operand)		**Roboter (Bit-Operand)**	
Ausgang	U3E0\G10000 bis U3E0\G10511	Eingang	Roboter-CPU Nr. 1 / 10000 bis 18191
	U3E0\G10512 bis U3E0\G11023		Roboter-CPU Nr. 2 / 10000 bis 18191
	U3E0\G11024 bis U3E0\G11535		Roboter-CPU Nr. 3 / 10000 bis 18191
Eingang	U3E1\G10000 bis U3E1\G10511	Ausgang	Roboter-CPU Nr. 1 / 10000 bis 18191
	U3E2\G10000 bis U3E2\G10511		Roboter-CPU Nr. 2 / 10000 bis 18191
	U3E3\G10000 bis U3E3\G10511		Roboter-CPU Nr. 3 / 10000 bis 18191

Tab. 10-4: *Gemeinsamer CPU-Speicher und Kompatibilität der Roboter-E/A-Signale*

10.2.3 SPS-Programm (Kontaktplan)

Beispiel▽ Die Taste „Roboterbetrieb freigeben“ des Steuerpults wird betätigt (X0 schaltet ein) und der Status „Betriebsrecht für Roboter aktiviert“ wird auf Y20 ausgegeben, wodurch die Signalleuchte „Roboterbetrieb freigegeben“ im Steuerpult einschaltet.

Das Multi-CPU-System besteht für die SPS-Funktion als erste CPU aus einer QnUD(H)CPU und für die Roboter-Funktion als zweite CPU aus einer Q172DRCPU.

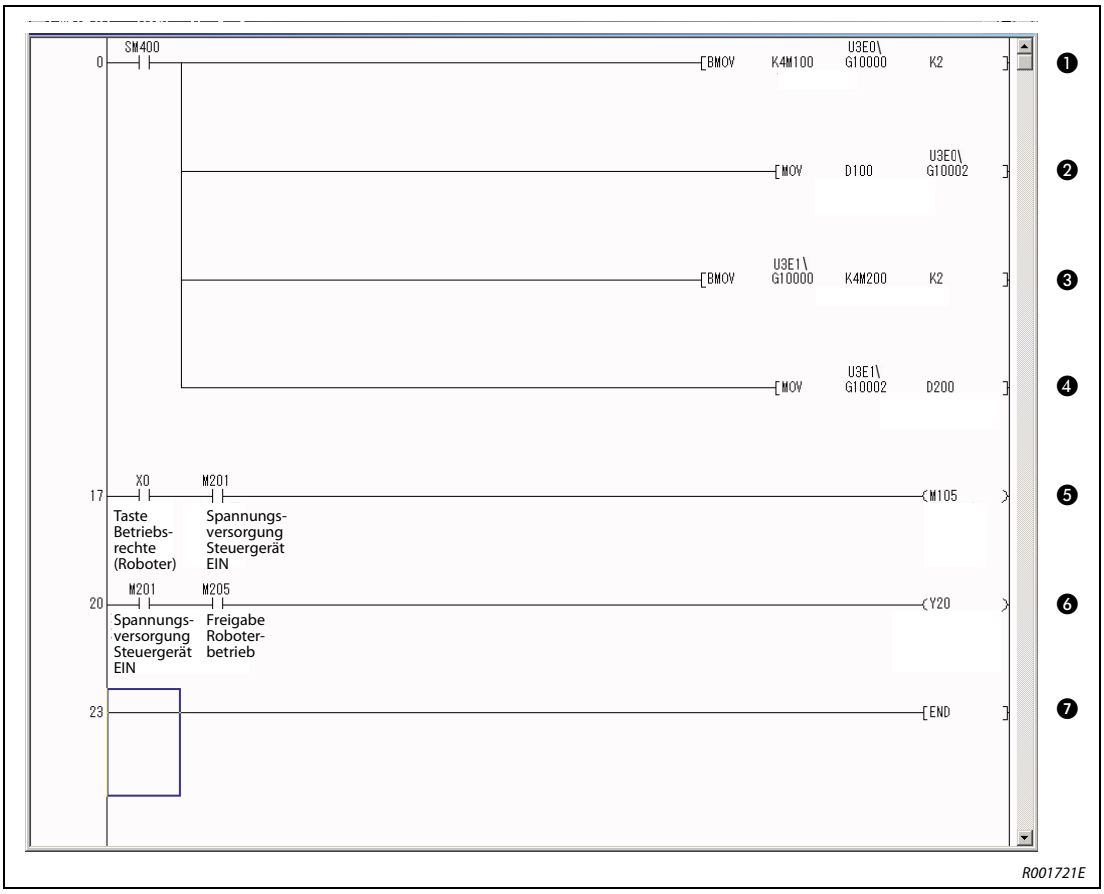


Abb. 10-1: *Kontaktplan für das Beispiel*

Nummer	Beschreibung
❶–❹	Die Merker M100 bis M131 werden in den gemeinsamen Speicher U3E0\G10000 und U3E0\G10001 geschrieben. Dies entspricht einer Eingabe von der SPS zum Roboter. Der gemeinsame Speicher U3E1\G10000 und U3E1\G10001 wird in die Bit-Operanden der Merker M200 bis M231 eingelesen, was einer Ausgabe vom Roboter zur SPS entspricht.
❶	Stopp-Eingabe
❷	Eingabe eines numerischer Werts in den Roboter (Register D100)
❸	Wartestatus
❹	Ausgabe eines numerischen Werts vom Roboter (Register D200)
❺–❼	Wenn der Eingang X0 einschaltet, schaltet auch M105 ein und das M105 entsprechende Bit 5 der SPS-Speicherstelle U3E0\G10000 schaltet ebenfalls ein. Daraufhin schaltet der Robotereingang 10005 ein und die von dem speziellen Eingangssignal zugewiesenen Betriebsrechte des Roboters werden aktiviert. Nach der Freigabe des Roboterbetriebs schaltet der über das parallele Ausgangssignal zugewiesene Roboterausgang 10005 ein, worauf das Bit 5 der SPS-Speicherstelle U3E1\G10000 ebenfalls einschaltet. Daraufhin schaltet der dem Bit 5 der SPS-Speicherstelle U3E1\G10000 zugewiesene Merker M205 ein, worauf dann der Ausgang Y20 einschaltet.
❺	Wenn der Eingang X0 und der Merker M201 ① gesetzt werden, schaltet M100 ein (Freigabeanforderung des Roboterbetriebs).
❻	Wenn die Merker M201 ① und M205 gesetzt werden, schaltet der Ausgang Y20 ein, worauf die Signalleuchte für die Betriebsrechte des Roboters einschaltet.
❼	Programmende

Tab. 10-5: *Beschreibung des Kontaktplans in Abb. 10-1*

① Beachten Sie bitte, dass in diesem Beispiel der Bit-Operand M201 (Speicherstelle U3E0\G10000, Bit 1, welches Roboterausgang 10001 entspricht) anzeigt, dass der Einschaltvorgang der Spannungsversorgung des Steuergerätes abgeschlossen ist (Ausgabe eines Signals, das die Empfangsbereitschaft für externe Eingangssignale anzeigt).

△

10.2.4 Zuordnung der speziellen Ein-/Ausgänge

Die folgende Tabelle zeigt die Zuordnung der speziellen Ein- und Ausgangssignale im Auslieferzustand (Werkseinstellung).

Parameter	Signalbezeichnung		Eingang	Ausgang	Operand G ①
	Eingang	Ausgang			
STOP	Stoppsignal (Zuordnung nicht änderbar)	Wartestatus aktiv	10000	10000	G10000
RCREADY	—	Spannungsversorgung des Steuergerätes eingeschaltet	—	10001	
ATEXTMD	—	Ausgangssignal externer Betrieb	—	10002	
TEACHMD	—	Ausgangssignal Teach-Modus	—	10003	
ATTOPMD	—	Ausgangssignal Automatikbetrieb	—	10004	
IOENA	Eingangssignal Betriebsrechte	Ausgangssignal Betriebsrechte	10005	10005	
START	Startsignal ②	Ausgangssignal Programm aktiv	10006	10006	
STOPSTS	—	Eingabe des Stoppsignals	—	10007	
SLOTINIT	Programme zurücksetzen ②	Ausgangssignal Programmwahl freigegeben	10008	10008	
ERRRESET	Fehler quittieren	Ausgangssignal Fehler	10009	10009	
SRVON	Servoversorgung einschalten ②	Servoversorgung eingeschaltet	10010	10010	
SRVOFF	Servoversorgung abschalten	Servos einschalten gesperrt	10011	10011	
CYCLE	Zyklischen Betrieb stoppen	Ausgangssignal zyklischer Betrieb gestoppt	10012	10012	
SAFEPOS	Eingangssignal Rückzugspunkt anfahren ②	Fährt den Rückzugspunkt an	10013	10013	
BATERR	—	Batteriespannung niedrig	—	10014	
OUTRESET	Allgemeine Ausgangssignale zurücksetzen ②	—	10015	—	
HLVLERR	—	Schwerer Fehler	—	10016	G10001
LLVLERR	—	Leichter Fehler	—	10017	
CLVLERR	—	Warnung	—	10018	
EMGERR	—	Ausgangssignal NOT-HALT	—	10019	
PRGSEL	Programmwahlsignal ②	—	10020	—	
OVRDSEL	Geschwindigkeitsübersteuerung auswählen	—	10021	—	
PRGOUT	Ausgabeanforderung Programmnummer	Ausgabe der Programmnummer	10022	10022	
LINEOUT	Ausgabeanforderung Zeilennummer	Ausgabe der Zeilennummer	10023	10023	
OVRDOUT	Ausgabeanforderung Geschwindigkeitsübersteuerung	Ausgabe der Geschwindigkeitsbeeinflussung	10024	10024	
ERROUT	Ausgabeanforderung Fehlernummer	Ausgabe der Fehlernummer	10025	10025	
—	—	—	—	—	
—	—	—	—	—	
—	—	—	—	—	
—	—	—	—	—	
—	—	—	—	—	
—	—	—	—	—	

Tab. 10-6: *Übersicht der Zuordnung der speziellen E/A bei Werkseinstellung (1)*

Parameter	Signalbezeichnung		Eingang	Ausgang	Operand G ①
	Eingang	Ausgang			
IODATA	Eingang 0 für numerische Eingabe	Ausgang 0 für numerische Ausgabe	10032	10032	G10002
	Eingang 1 für numerische Eingabe	Ausgang 1 für numerische Ausgabe	10033	10033	
	Eingang 2 für numerische Eingabe	Ausgang 2 für numerische Ausgabe	10034	10034	
	Eingang 3 für numerische Eingabe	Ausgang 3 für numerische Ausgabe	10035	10035	
	Eingang 4 für numerische Eingabe	Ausgang 4 für numerische Ausgabe	10036	10036	
	Eingang 5 für numerische Eingabe	Ausgang 5 für numerische Ausgabe	10037	10037	
	Eingang 6 für numerische Eingabe	Ausgang 6 für numerische Ausgabe	10038	10038	
	Eingang 7 für numerische Eingabe	Ausgang 7 für numerische Ausgabe	10039	10039	
	Eingang 8 für numerische Eingabe	Ausgang 8 für numerische Ausgabe	10040	10040	
	Eingang 9 für numerische Eingabe	Ausgang 9 für numerische Ausgabe	10041	10041	
	Eingang 10 für numerische Eingabe	Ausgang 10 für numerische Ausgabe	10042	10042	
	Eingang 11 für numerische Eingabe	Ausgang 11 für numerische Ausgabe	10043	10043	
	Eingang 12 für numerische Eingabe	Ausgang 12 für numerische Ausgabe	10044	10044	
	Eingang 13 für numerische Eingabe	Ausgang 13 für numerische Ausgabe	10045	10045	
	Eingang 14 für numerische Eingabe	Ausgang 14 für numerische Ausgabe	10046	10046	
	Eingang 15 für numerische Eingabe	Ausgang 15 für numerische Ausgabe	10047	10047	
HNDCNTL1	—	Handsteuersignal 900 für Hand	—	10048	G10003
	—	Handsteuersignal 901 für Hand	—	10049	
	—	Handsteuersignal 902 für Hand	—	10050	
	—	Handsteuersignal 903 für Hand	—	10051	
	—	Handsteuersignal 904 für Hand	—	10052	
	—	Handsteuersignal 905 für Hand	—	10053	
	—	Handsteuersignal 906 für Hand	—	10054	
	—	Handsteuersignal 907 für Hand	—	10055	
HNDSTS1	—	Handsensorsignal 900 für Hand	—	10056	
	—	Handsensorsignal 901 für Hand	—	10057	
	—	Handsensorsignal 902 für Hand	—	10058	
	—	Handsensorsignal 903 für Hand	—	10059	
	—	Handsensorsignal 904 für Hand	—	10060	
	—	Handsensorsignal 905 für Hand	—	10061	
	—	Handsensorsignal 906 für Hand	—	10062	
	—	Handsensorsignal 907 für Hand	—	10063	

Tab. 10-6: *Übersicht der Zuordnung der speziellen E/A bei Werkseinstellung (2)*

Parameter	Signalbezeichnung		Eingang	Ausgang	Operand G ①
	Eingang	Ausgang			
USRAREA	—	Über 8 Punkte festgelegter Arbeitsbereich 1	—	10064	G10004
	—	Über 8 Punkte festgelegter Arbeitsbereich 2	—	10065	
	—	Über 8 Punkte festgelegter Arbeitsbereich 3	—	10066	
	—	Über 8 Punkte festgelegter Arbeitsbereich 4	—	10067	
	—	Über 8 Punkte festgelegter Arbeitsbereich 5	—	10068	
	—	Über 8 Punkte festgelegter Arbeitsbereich 6	—	10069	
	—	Über 8 Punkte festgelegter Arbeitsbereich 7	—	10070	
	—	Über 8 Punkte festgelegter Arbeitsbereich 8	—	10071	

Tab. 10-6: *Übersicht der Zuordnung der speziellen E/A bei Werkseinstellung (3)*

① Adresse des gemeinsamen Operanden bei Einsatz der Multi-CPU
(Die Adresse ist von SPS-CPU-Seite aus gesehen)

② Betriebsrechte erforderlich

10.2.5 E/A-Adressanzahl der Modellreihen CRnQ-700 und CRn-500

E/A-Adressen	Modellreihe CRnQ-700	Modellreihe CRn-500		
	SPS-Link	Dezentrale E/A	CC-Link (optional)	PROFIBUS (optional)
Bit	8192/8192	256/256	Max. 126/126	Max. 3082/3082
			Bei Belegung von 4 Stationen	Gesamtanzahl E/A-Daten: 192 Worte (Max. Anzahl der Eingangs- oder Ausgangsdaten: 122 Worte)
Register	0/0	0/0	Max. 16/16 (Bei Belegung von 4 Stationen)	0/0

Tab. 10-7: *E/A-Adressvergleich*

10.3 Belegung der speziellen Ein- und Ausgänge

In folgender Tabelle sind die Funktionen aufgelistet, die den Ein-/Ausgängen zugewiesen werden können. Die Funktionen werden über Parameter den Signalnummern in der Reihenfolge Eingangssignalnummer/Ausgangssignalnummer zugewiesen. Die genaue Vorgehensweise zur Einstellung von Parametern finden Sie im Abschn. 3.18. Die Parameter können mit der Teaching Box im Menü zur Einstellung von Parametern oder über die optionale Software RT ToolBox2 eingestellt werden. Eine Verwendung der Funktionen ist nur dann möglich, wenn der MODE-Umschalter am Steuergerät auf „AUTOMATIC" eingestellt ist und zuvor das Eingangssignal Betriebsrechte (IOENA) eingeschaltet wurde.

Die Anzahl der verfügbaren Ein-/Ausgangssignale kann durch die optionalen parallelen Ein-/Ausgangsschnittstellen vergrößert werden.

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
RCREADY	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Spannungsversorgung des Steuergerätes eingeschaltet	Zeigt an, dass die Spannungsversorgung des Steuergerätes eingeschaltet ist und externe Signale empfangen werden können		10001	–1
ATEXTMD	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Ausgangssignal externer Betrieb	Zeigt an, dass der MODE-Umschalter am Steuergerät auf „AUTOMATIC" eingestellt ist Das Signal muss eingeschaltet sein, bevor die Funktionen der E/A genutzt werden können.		10002	–1
TEACHMD	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Ausgangssignal Teach-Modus	Zeigt an, dass der MODE-Umschalter am Steuergerät auf „MANUAL" eingestellt ist		10003	–1
ATTOPMD	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Ausgangssignal Automatikbetrieb	Zeigt an, dass der MODE-Umschalter am Steuergerät auf „AUTOMATIC" eingestellt ist		10004	–1
IOENA	Eingang	Eingangssignal Betriebsrechte	Anforderung der Betriebsrechte für eine externe Steuerung	H	10005	5
	Ausgang	Ausgangssignal Betriebsrechte	Zeigt an, dass der Betrieb über externe Signale freigegeben ist Die Betriebsrechte werden freigegeben, wenn das Eingangssignal eingeschaltet ist, der MODE-Umschalter auf „AUTOMATIC" eingestellt ist und kein anderes Gerät momentan über die Betriebsrechte verfügt.		10005	3

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (1)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
START (Betriebsrechte erforderlich)	Eingang	Startsignal	Startet die Programme im Multitasking-Betrieb Zum Starten eines bestimmten Programms muss das Programm über das Programmwahlsignal „PRGSEL“ und den numerischen Eingang „IODATA“ ausgewählt und anschließend über das Startsignal gestartet werden. Beachten Sie, dass die Programmnummer bei einer Freigabe des Parameters „PST“ über den numerischen Eingang „IODATA“ eingelesen und das entsprechende Programm unabhängig von der Programmwahl gestartet wird. Im Multitasking-Betrieb werden alle Programme gestartet. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht ausgeführt.	↑	10006	3
	Ausgang	Ausgangssignal Programm aktiv	Zeigt ein aktives Programm an Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht ausgeführt.		10006	0
STOP	Eingang	Stoppsignal	Stoppt die ausgeführten Programme (Programmplätze, deren Startbedingung auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht gestoppt) Die Eingangssignalnummer ist auf „0“ festgelegt. Im Multitasking- Betrieb werden alle Programme gestoppt. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht gestoppt. Mit dem Parameter INB kann beim Eingang zwischen Standard und Drahtbrucherkennung gewählt werden. HINWEIS: Verwenden Sie für alle sicherheitsrelevanten Stopps den NOT-HALT-Eingang.	H	10000 (keine Änderung möglich)	0 (keine Änderung möglich)
	Ausgang	Wartestatus aktiv	Zeigt an, dass die Abarbeitung des entsprechenden Programms vorübergehend unterbrochen worden ist Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht angezeigt.		10000	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (2)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
STOP2	Eingang	Stoppsignal	Stoppt die ausgeführten Programme Die Funktion entspricht der des STOP-Parameters. Im Gegensatz zum STOP-Parameter können jedoch die Signalnummern geändert werden.	H	–1	–1
	Ausgang	Wartestatus aktiv	Zeigt an, dass die Abarbeitung des entsprechenden Programms vorübergehend unterbrochen worden ist Die Funktion entspricht der des STOP-Parameters.		–1	–1
STOPSTS	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Eingabe des Stoppsignals	Zeigt an, dass das Stoppsignal eingegeben wurde (logische Addition aller Geräte)		10007	–1
SLOTINIT (Betriebsrechte erforderlich)	Eingang	Programme zurücksetzen	Setzt den Wartestatus der Programme und die Programme selbst zurück Das Zurücksetzen von Programmen ermöglicht eine Programmwahl. Im Multitasking- Betrieb werden alle Programmplätze zurückgesetzt. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht zurückgesetzt.	↑	10008	–1
	Ausgang	Ausgangssignal Programmwahl freigegeben	Zeigt an, dass die Programmwahl freigegeben ist Das Signal ist eingeschaltet, wenn das Programm weder ausgeführt wird, noch im Wartezustand ist. Im Multitasking-Betrieb ist das Signal eingeschaltet, wenn kein Programm ausgeführt wird oder sich im Wartezustand befindet. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht freigegeben.		10008	–1
ERRRESET	Eingang	Fehler quittieren	Quittiert den aktuellen Fehler	↑	10009	2
	Ausgang	Ausgangssignal Fehler	Zeigt an, dass ein Fehler aufgetreten ist		10009	2
SRVON (Betriebsrechte erforderlich)	Eingang	Servoversorgung einschalten	Schaltet die Servoversorgung für alle Servos ein	↑	10010	4,
	Ausgang	Servoversorgung eingeschaltet	Zeigt an, dass die Servoversorgung eingeschaltet ist		10010	1
SRVOFF	Eingang	Servoversorgung abschalten	Schaltet die Servoversorgung ab, das Einschalten der Servos wird gesperrt	H	10011	1
	Ausgang	Servos einschalten gesperrt	Zeigt an, dass das Einschalten der Servos gesperrt ist (Rückmeldung)		10011	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (3)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
AUTOENA	Eingang	Freigabe Automatikbetrieb	EIN: Automatikbetrieb freigegeben, AUS: Automatikbetrieb gesperrt Wird der Automatikbetrieb im gesperrten Zustand freigegeben, erfolgt die Fehlermeldung L5010. Die Funktion dient zur Verriegelung der E/A-Signale bei Bedienung über das Steuergerät.	H	–1	–1
	Ausgang	Ausgangssignal Automatikbetrieb freigegeben	Zeigt an, dass der Automatikbetrieb freigegeben ist		–1	–1
CYCLE	Eingang	Zyklischen Betrieb stoppen	Stoppt den zyklischen Betrieb	↑	10012	–1
	Ausgang	Ausgangssignal zyklischer Betrieb gestoppt	Zeigt an, dass der zyklische Betrieb gestoppt ist		10012	–1
MELOCK (Betriebsrechte erforderlich)	Eingang	Verriegelungssignal	Ein- bzw. Ausschalten des Verriegelungszustandes Der Signalpegel wird bei Programmauswahl auf „H“ gesetzt.	H	–1	–1
	Ausgang	Ausgangssignal Verriegelung aktiv	Zeigt an, dass der Mechanismus im verriegelten Zustand ist		–1	–1
SAFEPOS (Betriebsrechte erforderlich)	Eingang	Eingangssignal Rückzugspunkt anfahren	Anfahren des Rückzugspunkts Der im Parameter „JSAFE“ festgelegte Rückzugspunkt wird angefahren. Die Geschwindigkeit wird durch den Übersteuerungswert festgelegt. Achten Sie darauf, Kollisionen mit umliegenden Einheiten zu vermeiden.	↑	10013	–1
	Ausgang	Fährt den Rückzugspunkt an	Zeigt an, dass der Rückzugspunkt angefahren wird		10013	–1
BATERR	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Batteriespannung niedrig	Zeigt an, dass die Batteriespannung der Steuerung abgesunken ist. Der Ausgang BATERR schaltet ab, wenn die Steuerung nach dem Ersatz der Batterie wieder eingeschaltet wird. • Die aufsummierte Zeit, während der die Steuerung nicht eingeschaltet war, übersteigt 14600 Stunden. Der Ausgang BATERR schaltet ab, wenn die Betriebsdauer der Batterie zurück gesetzt wird.		10014	–1
OUTRESET (Betriebsrechte erforderlich)	Eingang	Allgemeine Ausgangssignale zurücksetzen	Zurücksetzen der allgemeinen Ausgangssignale	↑	10015	–1
	Ausgang	—	—		–1 (unwirksam)	–1 (unwirksam)
HLVLERR	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Schwerer Fehler	Zeigt an, dass ein schwerer Fehler aufgetreten ist		10016	–1
LLVLERR	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Leichter Fehler	Zeigt an, dass ein leichter Fehler aufgetreten ist		10017	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (4)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
CLVLERR	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Warnung	Zeigt eine Warnung an		10018	
EMGERR	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Ausgangssignal NOT-HALT	Zeigt den NOT-HALT-Status an		10019	–1
SnSTART (1 ≤ n ≤ 32) (Betriebsrechte erforderlich)	Eingang	Starteingang Programmplatz *n*	Startet das entsprechende Programm	↑	–1	–1
	Ausgang	Programm in Programmplatz *n* aktiv	Zeigt den aktiven Status jedes Programms an		–1	–1
SnSTOP (1 ≤ n ≤ 32) (Betriebsrechte erforderlich)	Eingang	Stoppeingang für Programmplatz *n*	Stoppt das entsprechende Programm	H	–1	–1
	Ausgang	Programm in Programmplatz *n* gestoppt	Zeigt an, dass das Programm vorübergehend gestoppt wurde		–1	–1
MnSRVOFF (1 ≤ n ≤ 3)	Eingang	Eingangssignal Servo AUS für Mechanismus *n*	Schaltet die Servoversorgung jedes Mechanismus aus und sperrt das Einschalten	H	–1	–1
	Ausgang	Ausgangssignal Servo EIN bei Mechanismus *n* gesperrt	Zeigt an, dass das Einschalten der Servoversorgung gesperrt ist (Rückmeldung)		–1	–1
MnSRVON (1 ≤ n ≤ 3) (Betriebsrechte erforderlich)	Eingang	Eingangssignal Servo EIN für Mechanismus *n*	Schaltet die Servoversorgung jedes Mechanismus ein	↑	–1	–1
	Ausgang	Ausgangssignal Servo EIN bei Mechanismus *n*	Zeigt an, dass die Servoversorgung eingeschaltet ist		–1	–1
MnMELOCK (1 ≤ n ≤ 3) (Betriebsrechte erforderlich)	Eingang	Eingangssignal Mechanismus *n* verriegeln	Schaltet die Verriegelung jedes Roboters aus oder ein	↑	–1	–1
	Ausgang	Ausgangssignal Mechanismus *n* verriegelt	Zeigt an, dass der Mechanismus im Verriegelungszustand ist		–1	–1
PRGSEL (Betriebsrechte erforderlich)	Eingang	Programmwahlsignal	Einlesen der numerischen Eingabe zur Programmwahl Das Signal sollte etwa 15 ms nach Ausgabe der numerischen Eingangsdaten „IODATA" für mindestens 15 ms an den Roboter ausgegeben werden.	↑	10020	–1
	Ausgang	—	—		–1 (unwirksam)	–1 (unwirksam)
OVRDSEL	Eingang	Geschwindigkeitsübersteuerung auswählen	Einlesen der numerischen Geschwindigkeitsübersteuerung Das Signal sollte etwa 15 ms nach Ausgabe der numerischen Eingangsdaten „IODATA" für mindestens 15 ms an den Roboter ausgegeben werden.	↑	10021	–1
	Ausgang	—	—		–1 (unwirksam)	–1 (unwirksam)

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (5)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
IODATA ③	Eingang	Eingang für numerische Eingabe (Start-Nr., End-Nr.)	Numerische Eingaben werden als binäre Werte eingelesen. • Programmnummer (Einlesen über PRGSEL) Ist der Parameter PST freigegeben, erfolgt das Einlesen mit dem Startsignal. • Geschwindigkeitsübersteuerung (Einlesen über OVRDSEL) Die Bitbreite kann beliebig gewählt werden. Die Genauigkeit sinkt bei Überschreitung des durch die Bitbreite möglichen Wertebereiches. Das Eingangssignal sollte 15 ms vor Eingabe des Signals PRGSEL oder anderer Einstellsignale anliegen.	H	10032 (Startbit) 10047 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
	Ausgang	Ausgang für numerische Ausgabe (Start-Nr., End-Nr.)	Numerische Eingaben werden als binäre Werte ausgegeben. • Programmnummer (Ausgabe über PRGOUT) • Geschwindigkeitsübersteuerung (Ausgabe über OVRDOUT) • Zeilennummer (Ausgabe über LINEOUT) • Fehlernummer (Ausgabe über ERROUT) Die Bitbreite kann beliebig gewählt werden. Die Genauigkeit sinkt bei Überschreitung des durch die Bitbreite möglichen Wertebereiches. Das Signal sollte 15 ms nach Eingabe der Programmnummer (PROGOUT) oder anderer Signale eingelesen werden.		10032 (Startbit) 10047 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
PRGOUT	Eingang	Ausgabeanforderung Programmnummer	Anforderung zur Ausgabe der Programmnummer über den numerischen Ausgang (IODATA) Nach Eingabe des Signals müssen mindestens 15 ms bis zum Einlesen des numerischen Aussgangssignals (IODATA) vergangen sein.	↑	10022	–1
	Ausgang	Ausgabe der Programmnummer	Zeigt an, dass die Programmnummer über den numerischen Ausgang ausgegeben wird		–1	–1
LINEOUT	Eingang	Ausgabeanforderung Schrittnummer	Anforderung zur Ausgabe der Schrittnummer über den numerischen Ausgang (IODATA) Nach Eingabe des Signals müssen mindestens 15 ms bis zum Einlesen des numerischen Ausgangssignals (IODATA) vergangen sein.	↑	10023	–1
	Ausgang	Ausgabe der Schrittnummer	Zeigt an, dass die Schrittnummer über den numerischen Ausgang ausgegeben wird		–1	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (6)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
OVRDOUT	Eingang	Ausgabeanforderung Geschwindigkeitsübersteuerung	Anforderung zur Ausgabe der Geschwindigkeitsübersteuerung über den numerischen Ausgang (IODATA) Nach Eingabe des Signals müssen mindestens 15 ms bis zum Einlesen des numerischen Ausgangssignals (IODATA) vergangen sein.	↑	10024	–1
	Ausgang	Ausgabe der Geschwindigkeitsbeeinflussung	Zeigt an, dass die Geschwindigkeitsübersteuerung über den numerischen Ausgang ausgegeben wird		–1	–1
ERROUT	Eingang	Ausgabeanforderung Fehlernummer	Anforderung zur Ausgabe der Fehlernummer über den numerischen Ausgang (IODATA) Nach Eingabe des Signals müssen mindestens 15 ms bis zum Einlesen des numerischen Ausgangssignals (IODATA) vergangen sein.	↑	10025	–1
	Ausgang	Ausgabe der Fehlernummer	Zeigt an, dass die Fehlernummer über den numerischen Ausgang ausgegeben wird		–1	–1
JOGENA (Betriebsrechte erforderlich)	Eingang	Freigabe JOG-Betrieb	Freigabe des JOG-Betriebs über externe Signale	H	–1	–1
	Ausgang	Freigabe JOG-Betrieb	Zeigt an, dass der JOG-Betrieb über externe Signale freigegeben ist		–1	–1
JOGM ③④	Eingang	2-Bit-Eingabe des JOG-Betriebs (Start-Nr., End-Nr.)	Festlegung des JOG-Betriebs 0/1/2/3/4 = Gelenk-, XYZ-, Kreis-, 3-Achsen-XYZ-, Werkzeug-, Werkstück-JOG-Betrieb HINWEIS: Der Werkstück-JOG-Betrieb steht nur in den folgenden Software-Version zur Verfügung: SD-Serie: ab Version P8 SQ-Serie: ab Version N8	H	–1 (Startbit) –1 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
	Ausgang	2-Bit-Ausgabe des JOG-Betriebs (Start-Nr., End-Nr.)	Ausgabe des aktuellen JOG-Betriebs		–1 (Startbit) –1 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
JOG+ ⑤	Eingang	JOG-Vorschub in positiver Richtung für 8 Achsen (Start-Nr., End-Nr.)	Festlegung des JOG-Betriebs in positiver Richtung beginnend mit der Startnummer Gelenk-JOG-Betrieb: J1, J2, J3, J4, J5, J6, J7, J8 XYZ-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 Kreis-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 3-Achsen-XYZ-JOG-Betrieb: X, Y, Z, J4, J5, J6 Werkzeug-JOG-Betrieb: X, Y, Z, A, B, C	H	–1	–1
	Ausgang	—	—		–1	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (7)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
JOG– ⑤	Eingang	JOG-Vorschub in negativer Richtung für 8 Achsen (Start-Nr., End-Nr.)	Festlegung des JOG-Betriebs in negativer Richtung beginnend mit der Startnummer Gelenk-JOG-Betrieb: J1, J2, J3, J4, J5, J6, J7, J8 XYZ-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 Kreis-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 3-Achsen-XYZ-JOG-Betrieb: X, Y, Z, J4, J5, J6 Werkzeug-JOG-Betrieb: X, Y, Z, A, B, C	H	–1	–1
	Ausgang	—	—		–1	–1
JOGWKNO ③④ Steht in folgenden Software-Versionen zur Verfügung: T/B: ab Ver. 1.3 SD-Serie: ab Ver. P8 SQ-Serie: ab Ver. N8	Eingang	Werkstückkoordinatennummer	Stellen Sie die Werkstückkoordinatennummer ein, die als Standard für der JOG-Werkstück-Betrieb verwendet werden soll Einstellbereich: 1–8 HINWEISE: Die Werkstückkoordinatennummer für den Standard-JOG-Werkstückbetrieb muss zwischen 1 und 8 eingestellt werden. Dieses Eingangssignal wird mit der ansteigenden Flanke des Signals JOGENA (Freigabe JOG-Betrieb) gelesen. Schalten Sie das Signal JOGENA einmal von AUS nach EIN, nachdem Sie die Werkstückkoordinatennummer geändert haben.	H	–1 (Startbit) –1 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
	Ausgang	Werkstückkoordinatennummer	Ausgabe der aktuell eingegebenen Werkstückkoordinatennummer		–1 (Startbit) –1 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
JOGNER (Betriebsrechte erforderlich)	Eingang	Fehler im JOG-Betrieb temporär ignorieren	Eingangssignal zum temporären Ignorieren von Fehlern, die im JOG-Betrieb nicht zurückgesetzt werden können Der Eingang kann ausschließlich für Mechanismus 1 geschaltet werden.	H	–1	–1
	Ausgang	Fehler im JOG-Betrieb temporär ignorieren	Zeigt das temporäre Ignorieren von Fehlern, die im JOG-Betrieb nicht zurückgesetzt werden können, an Der Ausgang kann ausschließlich von Mechanismus 1 geschaltet werden.		–1	–1
HNDCNTLn ($1 \leq n \leq 3$)	Eingang	—	—			
	Ausgang	Handsteuersignal für Hand Mechanismus *n* (Start-Nr., End-Nr.)	Ausgabe der Signalzustände der Handausgänge (n = 1) 900 bis 907 Ausgabe der Signalzustände der Handausgänge (n = 2) 910 bis 917 Ausgabe der Signalzustände der Handausgänge (n = 3) 920 bis 927 Beispiel: Zur Ausgabe der 4 Signale 900 bis 903 als allgemeine Ausgangssignale 3, 4, 5 und 6 setzen Sie HNDCNTL1 auf (3, 6)		HNDCNTL1 10048 (Startbit) 10055 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (8)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
HNDstSn (1 ≤ n ≤ 3)	Eingang	—	—		HNDstS1 10056 (Startbit) 10063 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
	Ausgang	Handsensorsignal für Hand Mechanismus *n* (Start-Nr., End-Nr.)	Ausgabe der Signalzustände der Handeingänge (n = 1) 900 bis 907 Ausgabe der Signalzustände der Handeingänge (n = 2) 910 bis 917 Ausgabe der Signalzustände der Handeingänge (n = 3) 920 bis 927 Beispiel: Zur Ausgabe der 4 Signale 900 bis 903 als allgemeine Ausgangssignale 3, 4, 5 und 6 setzen Sie HNDstS1 auf (3, 6)			
HANDENA Steht in folgenden Software-Versionen zur Verfügung: SD-Serie: ab Ver. P8 SQ-Serie: ab Ver. N8	Eingang	Freigabe Handsteuerung	Freigabe der Steuerung der Roboterhand über ein externes Signal freigegeben (1) / gesperrt (0) HINWEIS: Die Handausführung des Roboters steht während dem Automatikbetrieb zu Verfügung. Die Verriegelung des Roboters oder von externen Geräten erfolgt sicher und aus Sicherheitsgründen notwendig.	H	–1	–1
	Ausgang	Freigabe Handsteuerung	Ausgabe des Freigabestatus der Steuerung der Roboterhand freigegeben (1) / gesperrt (0) Wenn die Steuerung der Roboterhand über das Signal HANDENA ohne angeschlossene Teaching Box frei gegeben wird, schaltet dieses Ausgangssignal ein.		–1	–1
HANDOUT Steht in folgenden Software-Versionen zur Verfügung: SD-Serie: ab Ver. P8 SQ-Serie: ab Ver. N8	Eingang	Ausgabe Handsteuersignal (Start-Nr., End-Nr.)	Festlegung des externen Eingangssignalbereichs zur Steuerung der Roboterhand Das hier eingestellte Eingangssignal muss mit der Handsignaleinstellung in Parameter HANDTYPE übereinstimmen.	↑	–1	–1
	Ausgang	—	—		–1	–1
HNDERRn (1 ≤ n ≤ 3)	Eingang	Eingangssignal Fehler Hand *n*	Abfrage auf Handfehler Ein leichter Fehler (Fehlernummer 30) wird generiert.	H	–1	–1
	Ausgang	Ausgangssignal Fehler Hand *n*	Zeigt an, dass ein Handfehler aufgetreten ist Ein leichter Fehler (Fehlernummer 31) wird generiert.		–1	–1
AIRERRn (1 ≤ n ≤ 3)	Eingang	Luftdruck im Pneumatiksystem *n* fehlerhaft	Abfrage auf Pneumatikfehler	H	–1	–1
	Ausgang	Ausgabe Pneumatikfehler im System *n*	Zeigt an, dass ein Fehler im Pneumatiksystem aufgetreten ist		–1	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (9)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
USRAREA ⑥ Siehe auch Abschn. 9.9	Eingang	—	—		10064 (Startbit) 10071 (Stoppbit)	–1 (Startbit) –1 (Stoppbit)
	Ausgang	Über 8 Punkte festgelegter Arbeitsbereich (Start-Nr., End-Nr.)	Zeigt an, dass sich der Roboter im Arbeitsbereich befindet Die Ausgabe erfolgt der Reihe nach für die Bereiche 1, 2 und 3 beginnend mit der Startnummer. Die Festlegung des Bereiches erfolgt über die Parameter AREA1P1, AREA1P2 bis AREA8P1 und AREA8P2. Beispiele: für Bereich 1 –> USRAREA: 8, 8 für Bereich 1 und 2 –> USRAREA: 8, 9 Nicht erlaubt sind Einstellungen wie: USRAREA: 1, 1 (ungültig) USRAREA: 8, 1 (ungültig/ keine Fehlermeldung) USRAREA: 1, 8 (ungültig/ keine Fehlermeldung) USRAREA: 9, 8 (ungültig/ Fehlermeldung H6643)			
MnPTEXC (1 ≤ n ≤ 3)	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Warnmeldung Wartungsintervall abgelaufen	Zeigt an, dass das Wartungsintervall abgelaufen ist und Verschleißteile erneuert werden müssen	H	–1	–1
MnWUPENA (1 ≤ n ≤ 3) (Betriebsrechte erforderlich)	Eingang	Freigabe des Warmlaufbetriebs für Mechanismus *n*	Gibt den Warmlaufbetrieb für den festgelegten Mechanismus frei HINWEIS: Damit der Warmlaufbetrieb über dieses Eingangssignal freigegeben bzw. gesperrt werden kann, muss der Warmlaufbetrieb über den Parameter WUPENA freigegeben sein. Ansonsten bewirkt eine Eingabe des Signals keine Freigabe des Warmlaufbetriebs.	H	–1	–1
	Ausgang	Warmlaufbetrieb für Mechanismus *n* freigegeben	Zeigt an, dass der Warmlaufbetrieb freigegeben ist		–1	–1
MnWUPMD (1 ≤ n ≤ 3)	Eingang	—	—		–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Ausgangssignal Mechanismus *n* im Warmlaufbetrieb	Zeigt an, dass sich der Roboter im Warmlaufbetrieb befindet und mit reduzierter Geschwindigkeit verfahren wird	H	–1	–1
PSSLOT	Eingang	Festlegung Programmplatz	Die Nummer des Programmplatzes, der die Positionsdaten enthält, die der Anwender ausgeben möchte, wird geladen. Einstellbereich: 1–32 • Eine Änderung ist möglich, wenn der Eingangssignalstatus des Parameters PSOUT „0" ist.	H	–1 (Eingabe/Start-Nr.) –1 (Eingabe/End-Nr.) Maximale Länge: 6 Bit	
	Ausgang	Ausgabe Programmplatz	Der festgelegte Programmplatz wird ausgegeben		–1 (Ausgabe/Start-Nr.) –1 (Ausgabe/End-Nr.) Maximale Länge: 6 Bit	

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (10)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
PSTYPE	Eingang	Positionsdatentyp	Auswahl des Variablentyp für die Positionsdatenausgabe Einstellwerte: 0: Positionsvariable (P1, P10 o. Ä.) 1: Gelenkvariable (J1, J10 o. Ä.) • Eine Änderung ist möglich, wenn der Eingangssignalstatus des Parameters PSOUT „0" ist.	H	–1	–1
	Ausgang	Positionsdatentyp	Ausgabe des ausgewählten Positionsdatentyps. Ausgabewerte: 0: Positionsvariable (P1, P10 o. Ä.) 1: Gelenkvariable (J1, J10 o. Ä.)		–1	–1
PSNUM	Eingang	Positionsnummer	Die Nummer der Positions- (P) oder Gelenkvariablen (J) für die Positionsdatenausgabe wird festgelegt. Einstellbereich: 0–65535 (P0–P65535 oder J0–J65535) Beispiel: Stellen Sie den Wert 100 ein, wenn Sie die Daten der Positionsvariablen P100 benötigen. Die zugehörige Signalnummer muss zwischen Eingabe/Start-Nr und der Eingabe/End-Nr. liegen. Für die Signalnummer können maximal 16 Bit genutzt werden, wobei Positionsdaten bis „P65535" möglich sind. Beachten Sie, dass die Positionsvariable „P001" ungültig ist. • Eine Änderung ist möglich, wenn der Eingangssignalstatus des Parameters PSOUT „0" ist.	H	–1 (Eingabe/Start-Nr.) –1 (Eingabe/End-Nr.) Maximale Länge: 16 Bit	
	Ausgang	Positionsnummer	Ausgabe Positionsnummer.		–1 (Ausgabe/Start-Nr.) –1 (Ausgabe/End-Nr.) Maximale Länge: 16 Bit	
PSOUT	Eingang	Anforderung Positionsdatenausgabe	Die Datenausgabe der Positionsnummer für den festgelegten Programmplatz wird festgelegt Die Positionsdaten werden aktualisiert, wenn dieses Signal einschaltet. Obwohl dieser Eingang zustandsgesteuert ist (H), werden die Positionsdaten nicht ständig aktualisiert, wenn dieses Signal eingeschaltet bleibt. Die Aktualisierung erfolgt nur, wenn dieses Signal einschaltet (1). 0: Die Positionsdatenausgabe ist nicht notwendig oder die Positionsnummer wird gerade festgelegt 1: Anforderung der Positionsdatenausgabe	H	–1	–1
	Ausgang	Positionsdatenausgabe erfolgt	Zeigt an, dass die Positionsdaten ausgegeben wurden. 0: Positionsdaten noch nicht ausgegeben 1: Positionsdaten wurden ausgegeben		–1	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (11)*

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ①	Werkseinstellung ②	
					CRnQ	CRnD
PSPOS	Eingang	—	—	—	–1 (unwirksam)	–1 (unwirksam)
	Ausgang	Positionsdaten	Die festgelegten Positionsdaten werden über Signale ausgegeben, die aus 32 Bits für 8 Achsen plus 2 Elemente (Positionsmerker) bestehen. Die Signale werden von den Signalnummern abgeleitet, die mit diesem Parameter festgelegt werden (320 Bits werden verwendet). Einstellbereiche: • Modellreihe CRnD-700 2000–3632: Profibus 6000–7728: CC-Link • Modellreihe CRnQ-700 10000–17872: gemeinsame Multi-CPU-Operanden Die Einheit für jeden Positionsdatenwert, bestehend aus einem 32-Bit-Signal, ist µm: 1 µm = 10^{-3} mm oder 10^{-3} Grad Beachten Sie, dass der Wert als ganze Zahl mit Vorzeichen ausgegeben wird (-2^{31} bis ($+2^{31}$ - 1)) Positionsmerker werden mit ihrem eigenen Wert ohne Umwandlung in µm ausgegeben. • Positionsvariablen: X, Y, Z, A, B, C, L1, L2, FL1, FL2 • Gelenkvariablen: J1, J2, J3, J4, J5, J6, J7, J8 Tritt in einem Programmplatz ein Fehler auf, wird „0x7FFFFFFF“ ausgegeben, bei einem Fehler in der Positionsnummer wird „0x80000000“ ausgegeben. Beachten Sie dazu auch das Zeitablaufdiagramm 5 bei externer Steuerung, das in Abb. 10-7 gezeigt wird.		–1	–1
TMPOUT	Eingang	Temperaturabfrage	Die Temperatur im Steuergerät wird über den numerischen Ausgang IODATA ausgegeben. Warten Sie nach Eingabe des Signals ca. 15 ms, bevor Sie die Daten am Ausgang IODATA auslesen.	↑	–1	–1
	Ausgang	Temperaturausgabe	Ausgabe der Temperatur im Steuergerät über den numerischen Ausgang IODATA	—	–1	–1

Tab. 10-8: *Spezielle Parameter für Ein-/Ausgänge (12)*

Beachten Sie auch die Fußnoten auf der nächsten Seite.

① Eingang: „H“ bedeutet, die Funktion ist aktiv, wenn das externe Signal eingeschaltet ist und inaktiv, wenn das externe Signal ausgeschaltet ist. Das Signal muss mindestens 15 ms eingeschaltet sein.
Eingang: „ ↑ “ bedeutet, die Funktion ist aktiv, wenn das externe Signal vom AUS- in den EIN-Zustand wechselt. Die aktivierte Funktion bleibt auch nach einem Wechsel des externen Signals in den AUS-Zustand erhalten.
Bsp.: Bei einer Einstellung der Startbedingung im Programmplatzparameter auf START (CYC, ERROR usw.) wird das Programm mit steigender Flanke gestartet. Das Programm wird nicht mit steigender Flanke gestoppt.

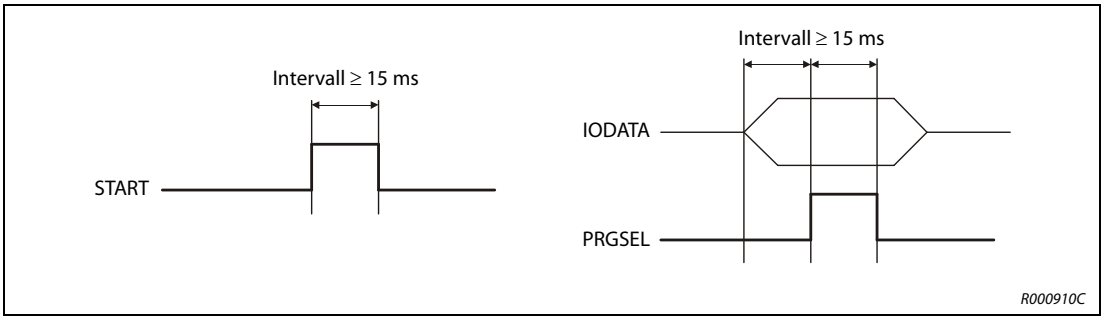


Abb. 10-2: *Signaldauer*

② Die Werkseinstellung „–1“ bedeutet, dass die Funktion nicht aktiviert ist.

③ Die Eingabe erfolgt in der Reihenfolge: Eingangsstartnummer, Eingangsendnummer, Ausgangsstartnummer, Ausgangsendnummer. Geben Sie bei einer Ein-/Ausgabe eines aktuellen Wertes die Start- und Endnummer als binären Wert an. Dabei entspricht die Startnummer dem niederwertigen, die Endnummer dem höherwertigen Bit. Setzen Sie nur die zur Einstellung notwendigen Werte. Stehen z.B. bei einer Programmwahl nur die Programme 1 bis 6 zur Auswahl, reichen zur Darstellung 3 Bits. Es können bis zu 16 Bits gesetzt werden.

Beispiele ▽ Die Zuweisung des Starteingangssignals an Eingang 10016 und des Ausgangssignals „Programm aktiv“ an Ausgang 10026 erfolgt über:
Parameter START = [10016, 10026]

Die Zuweisung von 4 Bits der numerische Eingabe an die Eingänge 10027 bis 10030 und von 5 Bits der numerischen Ausgabe an die Ausgänge 10027 bis 10031 erfolgt über:
Parameter IODATA = [10027, 10030, 10027, 10031]

△

④ Die Eingabe erfolgt in der Reihenfolge: Eingangsstartnummer, Eingangsendnummer, Ausgangsstartnummer, Ausgangsendnummer. Geben Sie die Start- und Endnummer als binären Wert an. Dabei entspricht die Startnummer dem niederwertigen, die Endnummer dem höherwertigen Bit. Setzen Sie nur die zur Einstellung notwendigen Werte.

⑤ Die Eingabe erfolgt in der Reihenfolge: Eingangsstartnummer, Eingangsendnummer. Über die Startnummer wird die Achse J1/X festgelegt und über die Endnummer können Achsen bis zu J8/L2 festgelegt werden.

⑥ Die Eingabe erfolgt in der Reihenfolge: Ausgangsstartnummer, Ausgangsendnummer. Über die Startnummer wird der Bereich 1, über die Endnummer maximal der Bereich 32 festgelegt. Die Festlegung zweier Benutzerbereiche erfolgt über zwei Bits. Es können maximal 32 Bits gesetzt werden.

Freigabe der zugewiesenen Eingangssignale

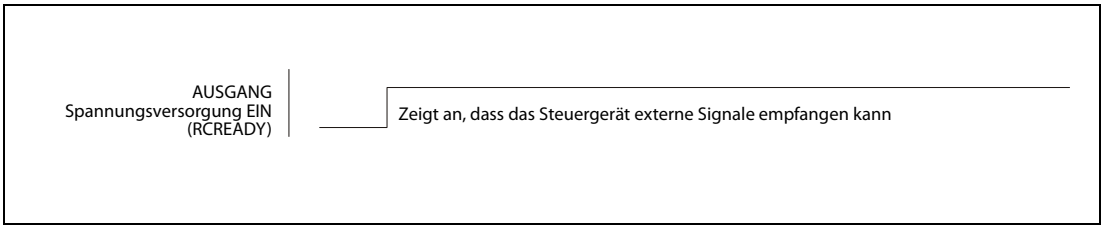
Die Gültigkeit eines anliegenden und zugewiesenen Eingangssignals hängt vom Betriebszustand des Roboters ab.

Parameter	Bezeichnung	Gültigkeit
SLOTINIT	Programme zurücksetzen	Keine Funktion während des Betriebs (bei Ausgabe des START-Signals)
SAFEPOS	Eingangssignal Ersatzposition anfahren	
OUTRESET	Allgemeine Ausgangssignale zurücksetzen	
PRGSEL	Programmwahlsignal	
MnWUPENA	Eingangssignal zur Freigabe des Warmlaufbetriebs für Mechanismus n	
START SnSTART ($1 \leq n \leq 32$)	Startsignal	Funktion nur bei Ausgabe des IOENA-Signals
SLOTINIT	Programme zurücksetzen	
SRVON MnSRVON ($1 \leq n \leq 3$)	Servoversorgung einschalten	
MELOCK MnMELOCK ($1 \leq n \leq 3$)	Verriegelungssignal	
SAFEPOS	Eingangssignal Ersatzposition anfahren	
PRGSEL	Programmwahlsignal	
OVRDSEL	Geschwindigkeitsübersteuerung auswählen	
JOGENA	Freigabe JOG-Betrieb	
MnWUPENA	Eingangssignal zur Freigabe des Warmlaufbetriebs für Mechanismus n	
START	Startsignal	Keine Funktion bei Eingabe des Stoppsignals (bei Ausgabe des STOPSTS-Signals)
SLOTINIT	Programme zurücksetzen	
SAFEPOS	Eingangssignal Ersatzposition anfahren	
JOGENA	Freigabe JOG-Betrieb	
SRVON	Servoversorgung einschalten	Keine Funktion bei eingeschaltetem SRVOFF-Signal
MELOCK	Verriegelungssignal	Funktion nur im Programmauswahlmodus (bei Ausgabe des SLOTINIT-Signals)
PRGSEL	Programmwahlsignal	Keine Funktion während der Unterbrechung (das Ausgangssignal des Parameters STOP ist hier eingeschaltet (Wartestatus aktiv).)

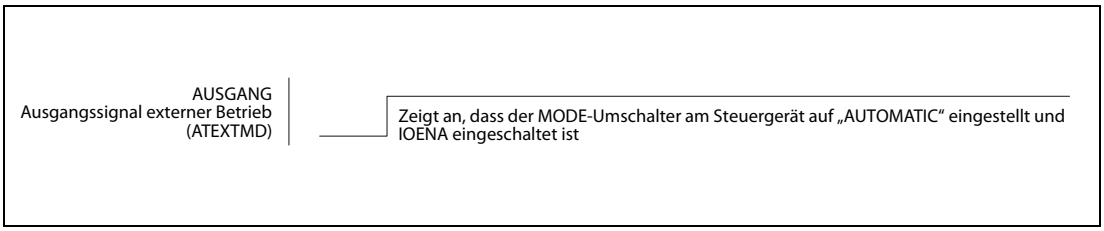
Tab. 10-9: *Gültigkeit der Eingangssignale*

Zeitablaufdiagramme externer Signale

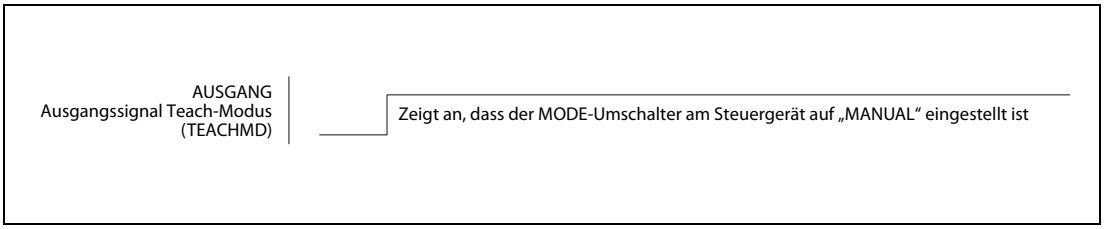
- RCREADY (Ausgang Spannungsversorgung des Steuergerätes eingeschaltet)



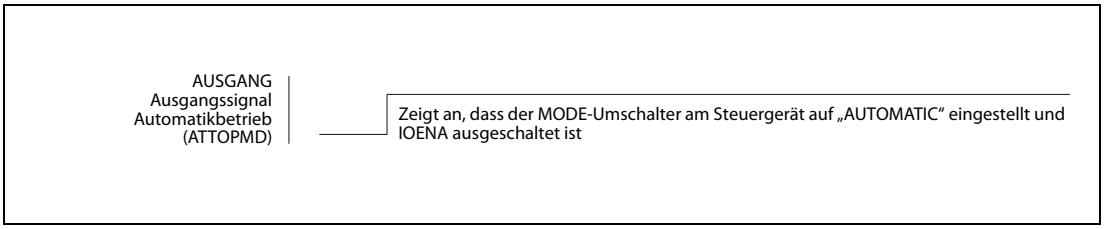
- ATEXTMD (Ausgang externer Betrieb)


AUSGANG
Ausgangssignal externer Betrieb
(ATEXTMD)
Zeigt an, dass der MODE-Umschalter am Steuergerät auf „AUTOMATIC“ eingestellt und IOENA eingeschaltet ist

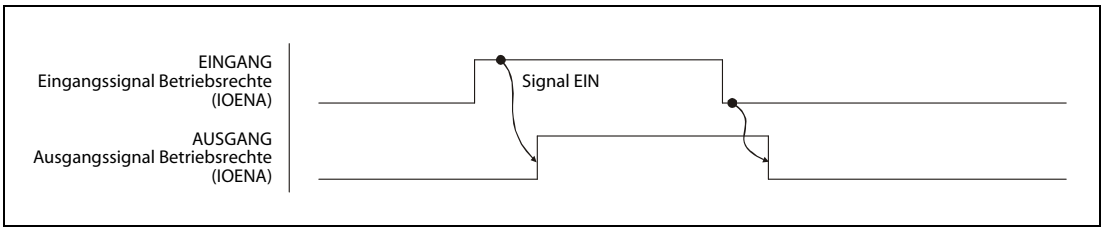
- TEACHMD (Ausgangssignal Teach-Modus)



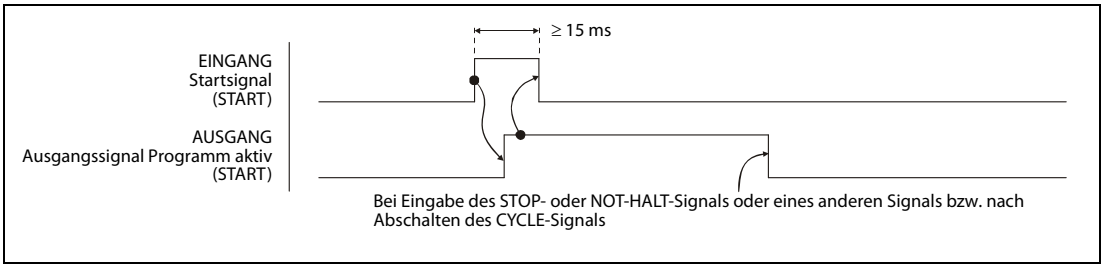
- ATTOPMD (Ausgang Automatikbetrieb)



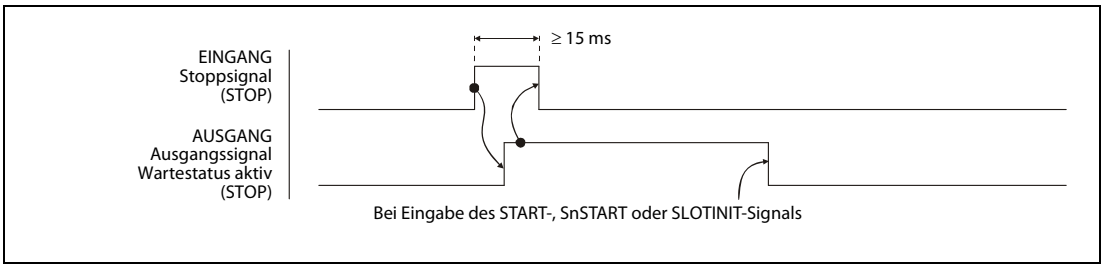
- IOENA (Eingang Betriebsrechte/Ausgang Betriebsrechte)



● START (Eingang Startsignal/Ausgang Programm aktiv)

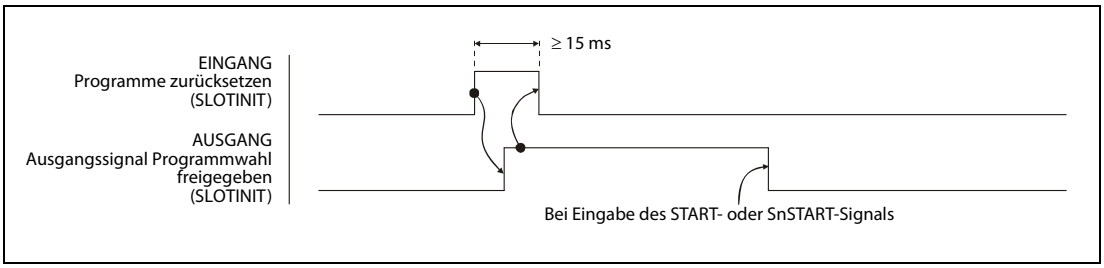


● STOP (Eingang Stoppsignal/Ausgang Wartestatus aktiv)

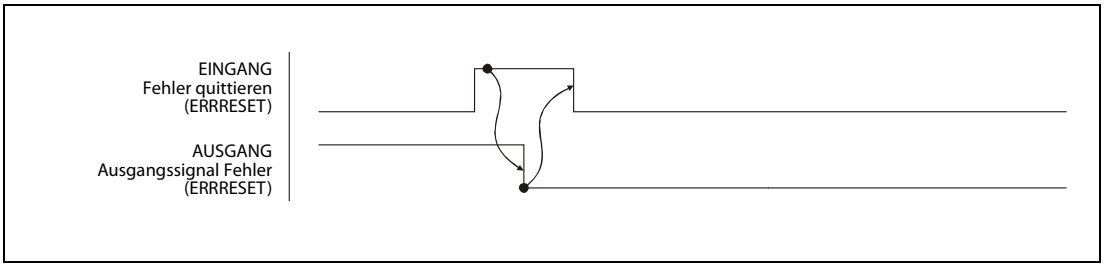


● STOPSTS (Ausgang Eingabe des Stoppsignals)

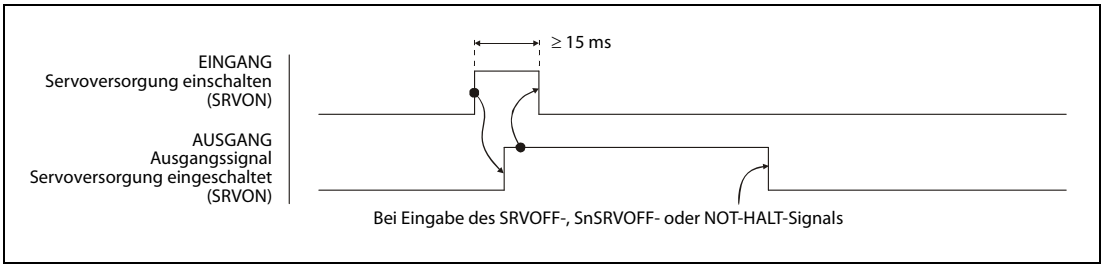
● SLOTINIT (Eingang Programme zurücksetzen/Ausgang Programmwahl freigegeben)



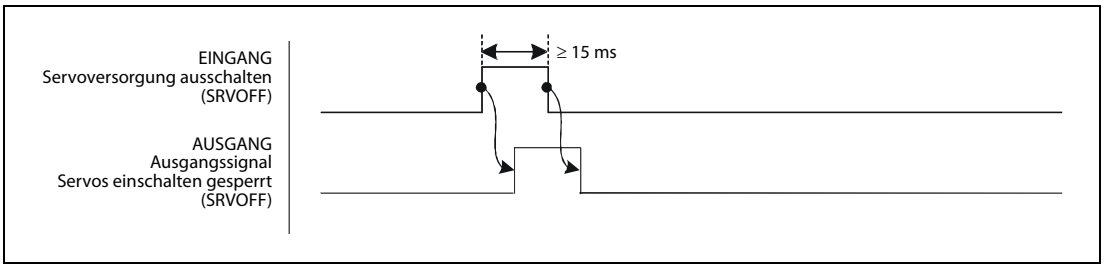
● ERRRESET (Fehler quittieren/Ausgang Fehler)



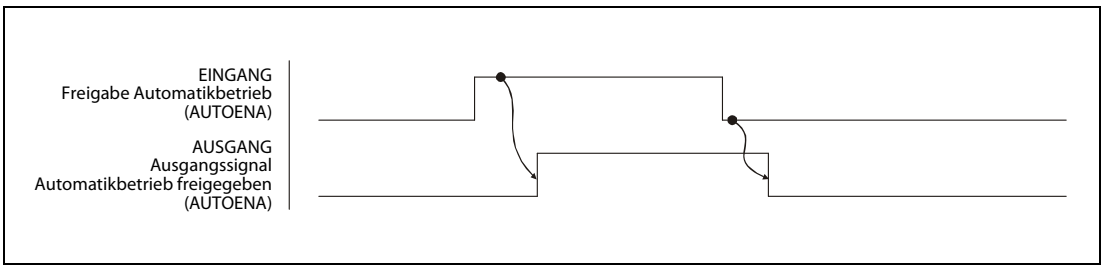
- SRVON (Eingang Servoversorgung einschalten/Ausgang Servoversorgung eingeschaltet)



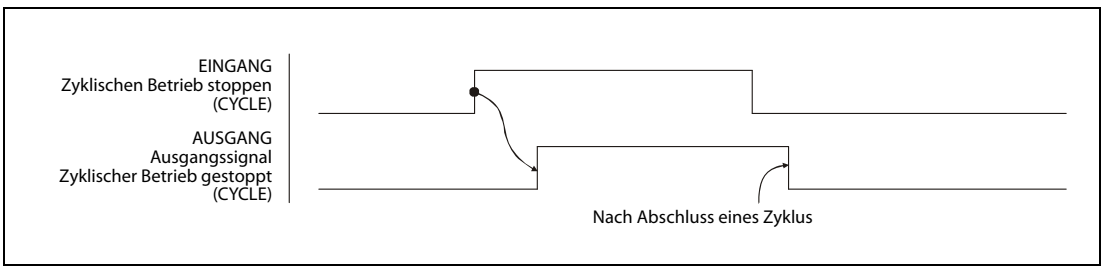
- SRVOFF (Eingang Servoversorgung ausschalten/Ausgang Servos einschalten gesperrt)



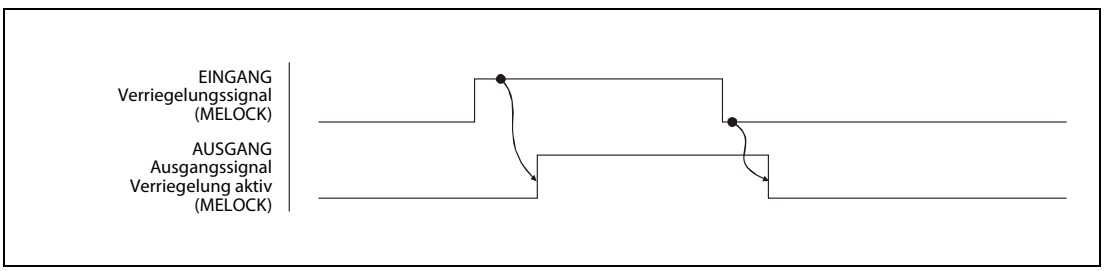
- AUTOENA (Eingang Freigabe Automatikbetrieb/Ausgang Automatikbetrieb freigegeben)



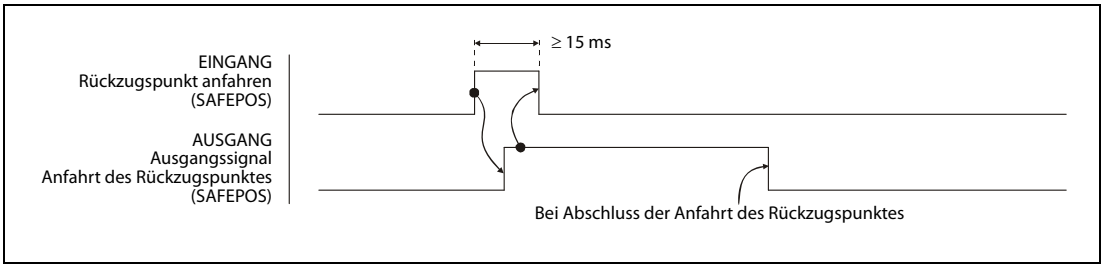
- CYCLE (Eingang zyklischen Betrieb stoppen/Ausgang zyklischer Betrieb gestoppt)



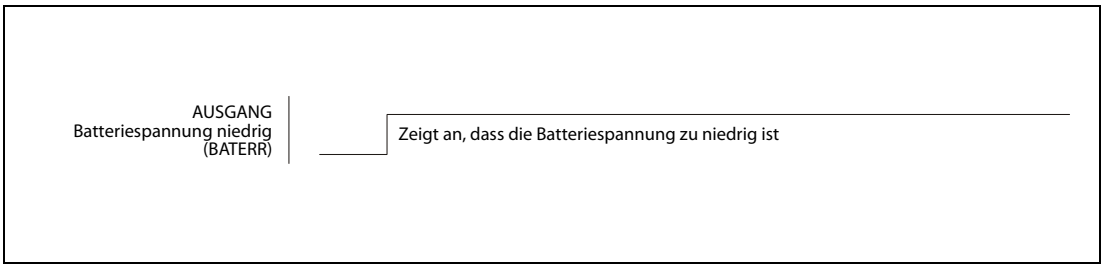
- MELOCK (Eingang Verriegelungssignal/Ausgang Verriegelung aktiv)



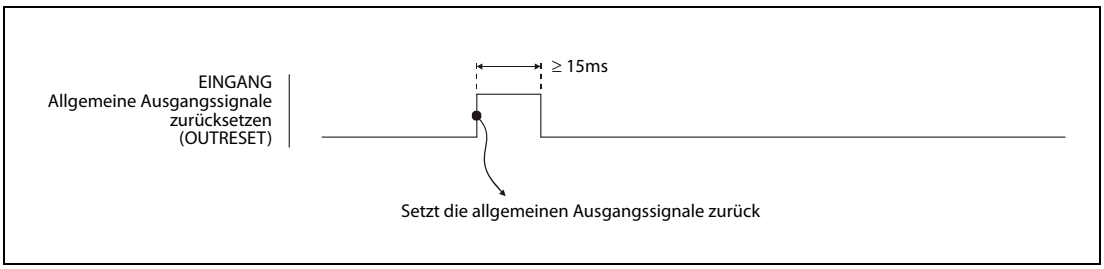
● SAFEPOS (Eingang Rückzugspunkt anfahren/Ausgang Anfahrt des Rückzugspunktes)



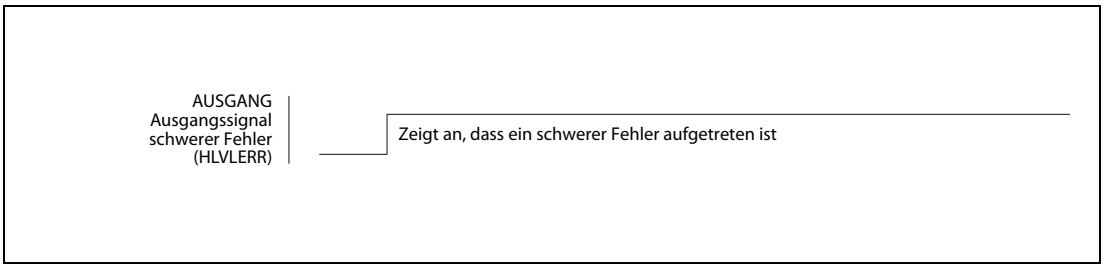
● BATERR (Ausgang Batteriespannung niedrig)



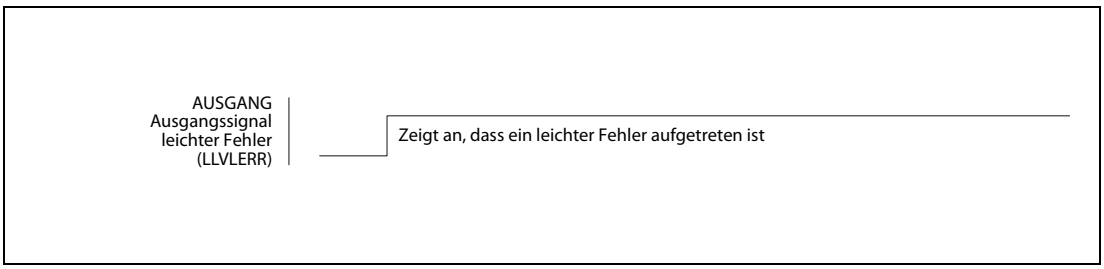
● OUTRESET (Eingang zum Zurücksetzen der allgemeinen Ausgangssignale)



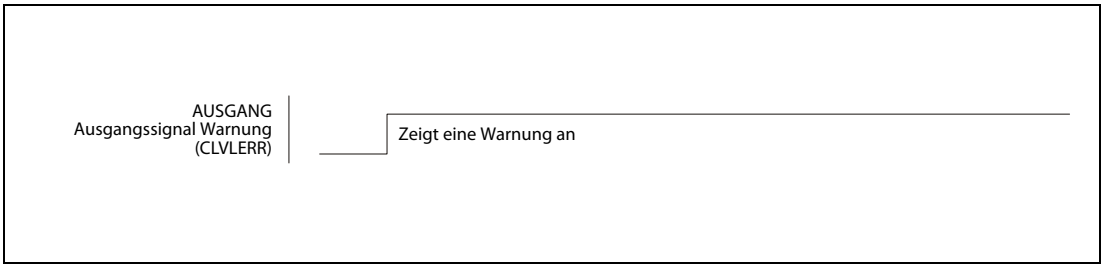
● HLVLERR (Ausgang zur Anzeige schwerer Fehler)



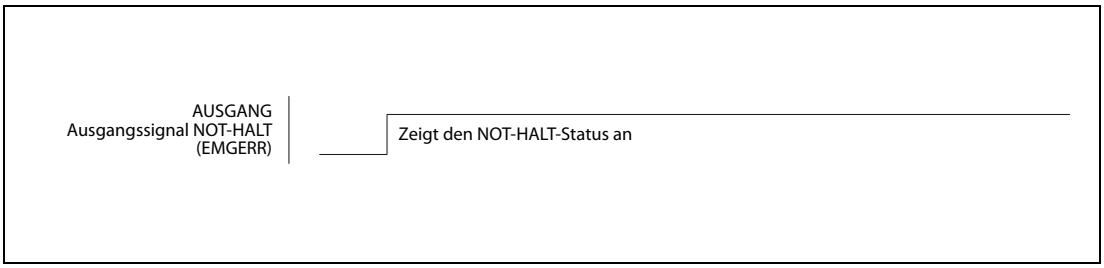
● LLVLERR (Ausgang zur Anzeige leichter Fehler)



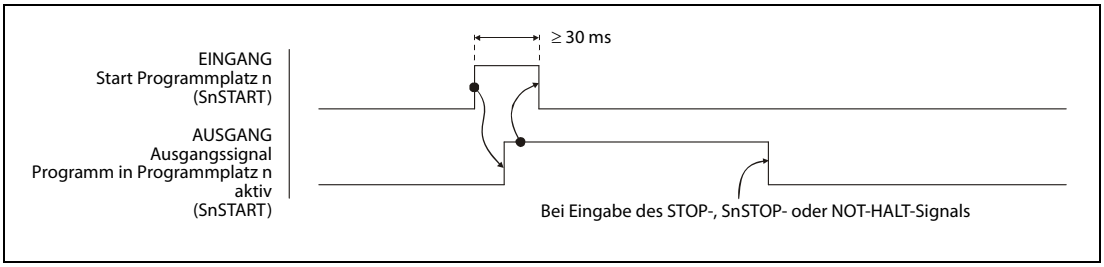
- CLVLERR (Ausgang zur Anzeige von Warnungen)



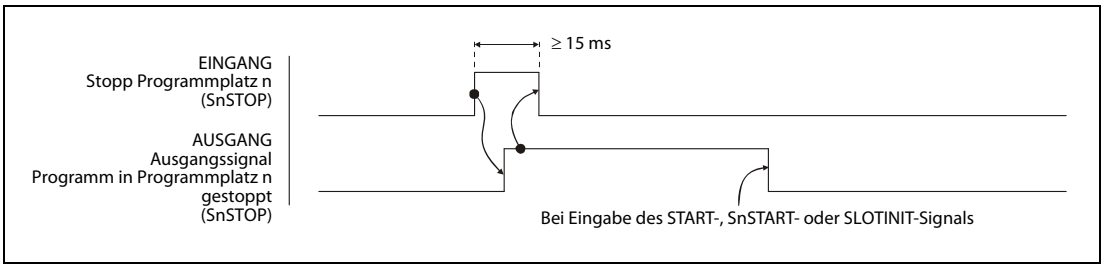
- EMGERR (Ausgang NOT-HALT)



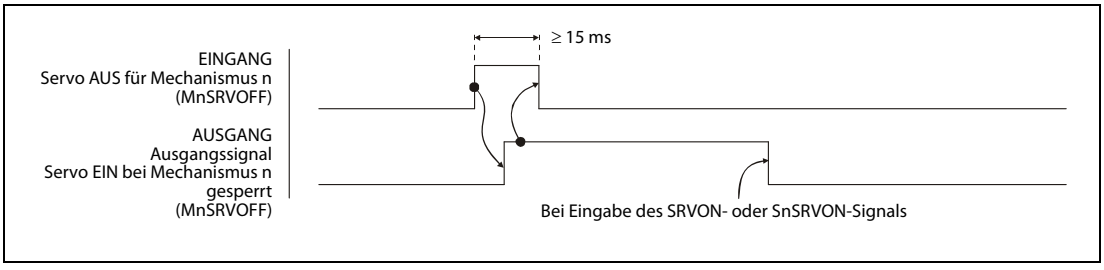
- SnSTART (Starteingang Programmplatz n/Ausgang Programm in Programmplatz n aktiv)



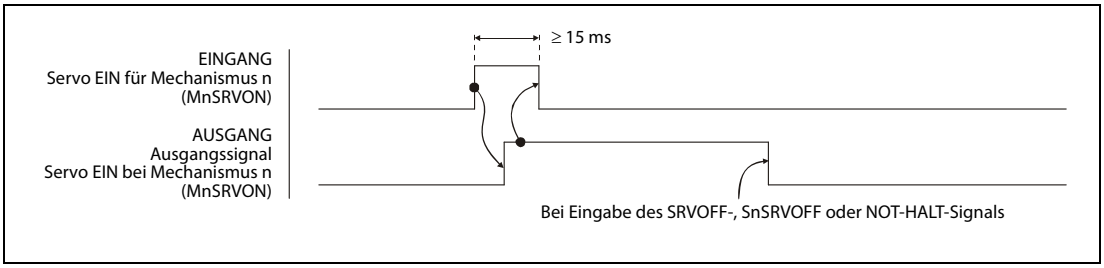
- SnSTOP (Stoppeingang Programmplatz n/Ausgang Programm in Programmplatz n gestoppt)



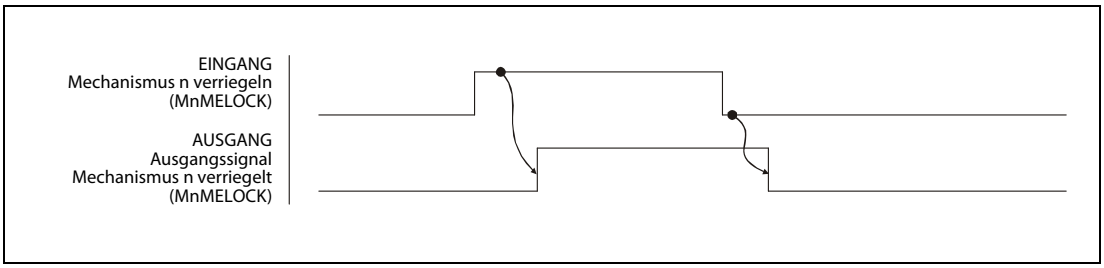
- MnSRVOFF (Eingang Servo AUS für Mechanismus n/Ausgang Servo EIN bei Mechanismus n gesperrt)



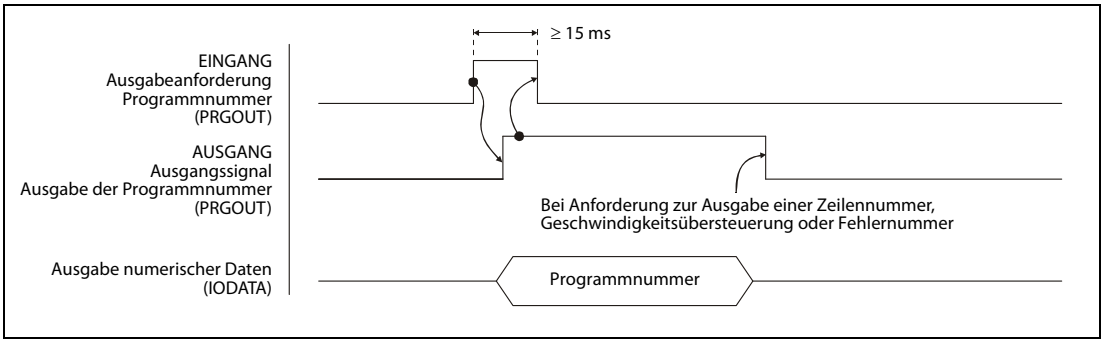
- MnSRVON (Eingang Servo EIN für Mechanismus n/Ausgang Servo EIN bei Mechanismus n)



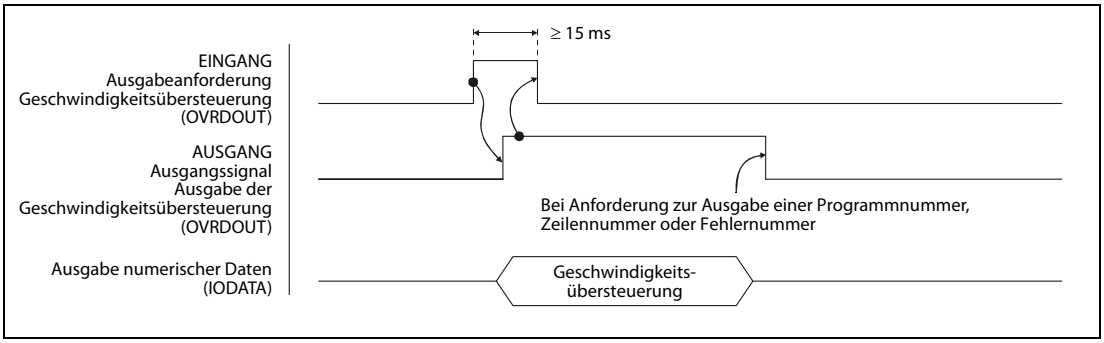
- MnMELOCK (Eingang Mechanismus n verriegeln/Ausgang Mechanismus n verriegelt)



- PRGSEL (Eingang Programmwahlsignal)
  Der Eingang wird zusammen mit dem Eingang für numerische Daten „IODATA“ verwendet.

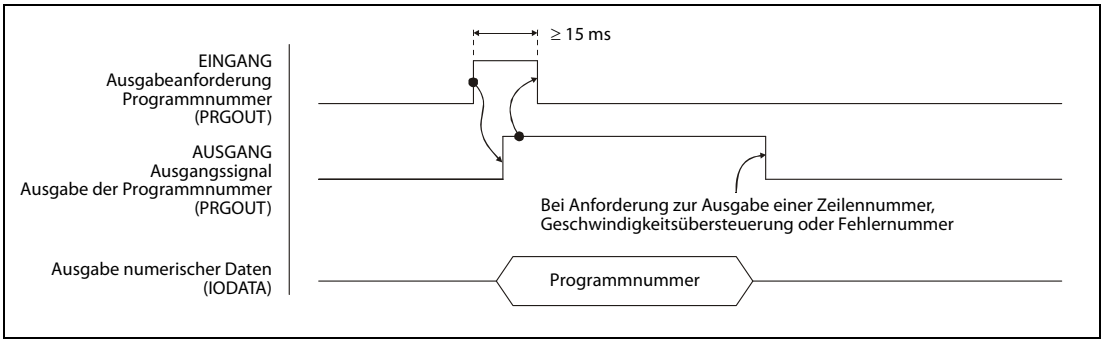


- OvrdSEL (Eingang Geschwindigkeitsübersteuerung auswählen)
  Der Eingang wird zusammen mit dem Eingang für numerische Daten „IODATA“ verwendet.

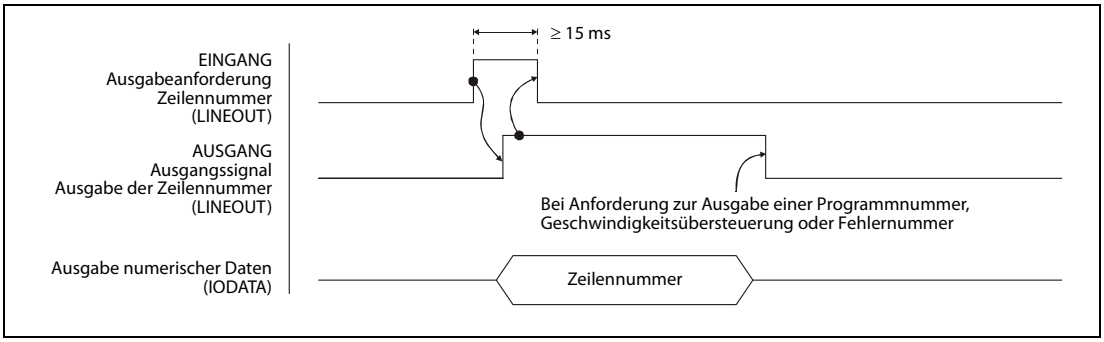


- IODATA (Eingang für numerische Eingabe/Ausgang für numerische Ausgabe)
  IODATA wird zusammen mit dem Signal PRGSEL, OvrdSEL, PRGOUT, LINEOUT, OVRDOUT oder ERROUT verwendet.

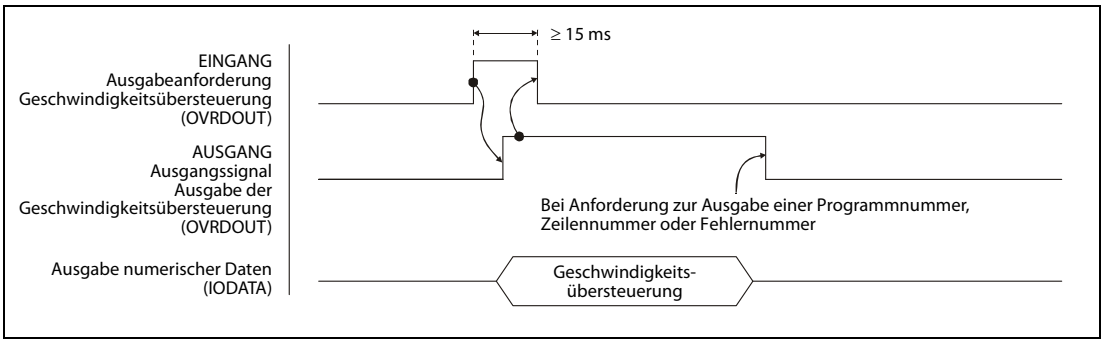
- PRGOUT (Eingang Ausgabeanforderung Programmnummer/Ausgabe der Programmnummer)
  PRGOUT wird zusammen mit dem Ausgang für numerische Daten „IODATA“ verwendet.



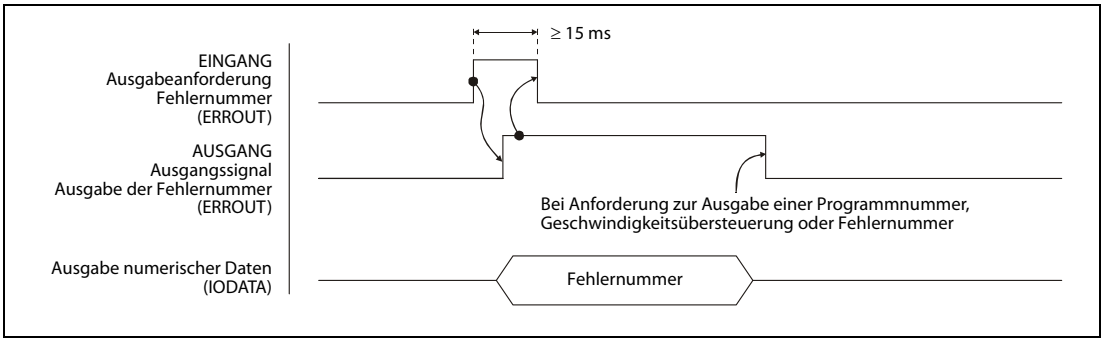
- LINEOUT (Eingang Ausgabeanforderung Zeilennummer/Ausgabe der Zeilennummer)
  LINEOUT wird zusammen mit dem Ausgang für numerische Daten „IODATA“ verwendet.



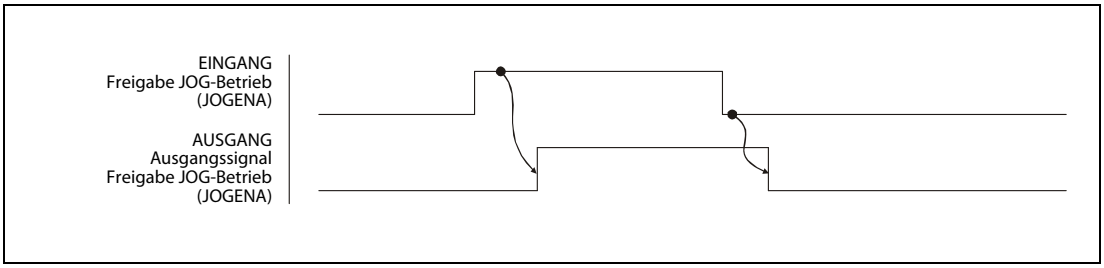
- OVRDOUT (Eingang Ausgabeanforderung Geschwindigkeitsübersteuerung/Ausgabe der Geschwindigkeitsübersteuerung)
  OVRDOUT wird zusammen mit dem Ausgang für numerische Daten „IODATA“ verwendet.



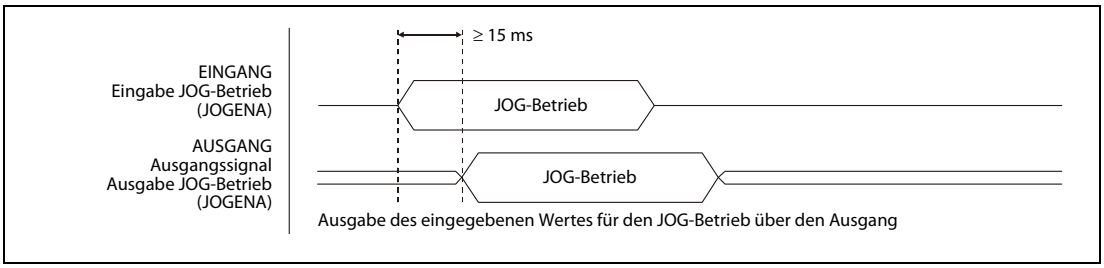
- ERROUT (Eingang Ausgabeanforderung Fehlernummer/Ausgabe der Fehlernummer)
  ERROUT wird zusammen mit dem Ausgang für numerische Daten „IODATA“ verwendet.



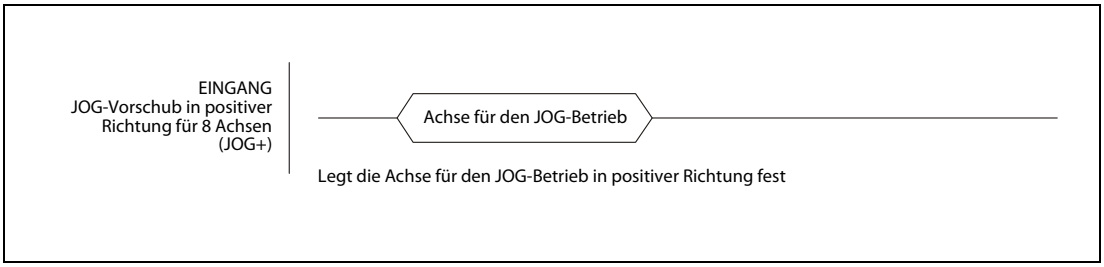
● JOGENA (Eingang Freigabe JOG-Betrieb/Freigabe JOG-Betrieb)



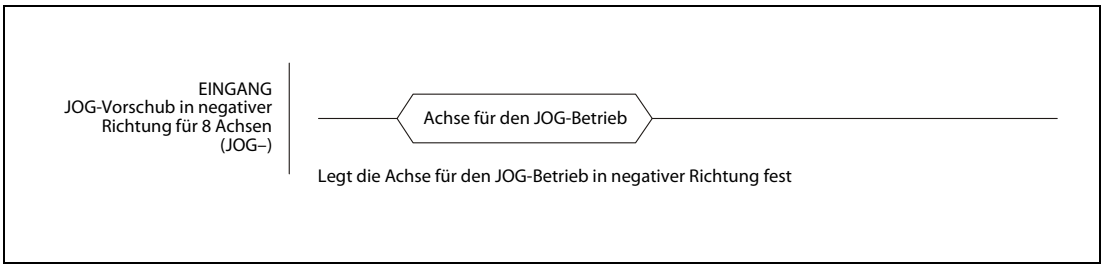
● JOGM (Eingang 2-Bit-Eingabe des JOG-Betriebs/2-Bit-Ausgabe des JOG-Betriebs)



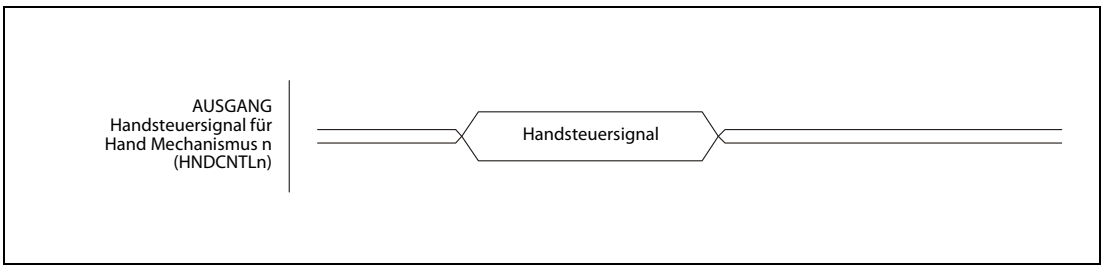
● JOG+ (Eingang JOG-Vorschub in positiver Richtung für 8 Achsen)



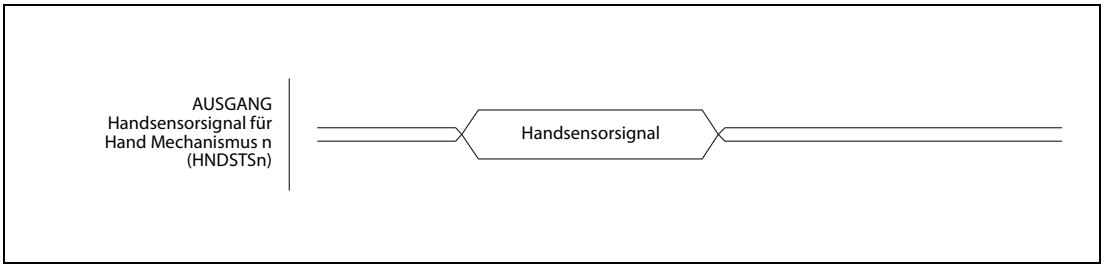
● JOG– (Eingang JOG-Vorschub in negativer Richtung für 8 Achsen)



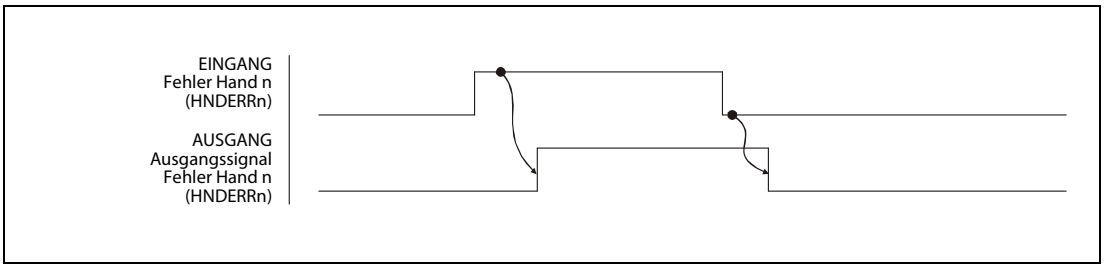
● HNDCNTLn (Handsteuersignal für Hand Mechanismus n)



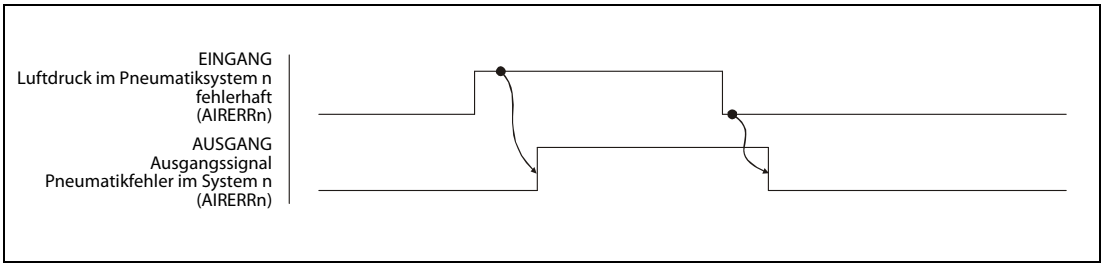
- HNDSTSn (Handsensorsignal für Hand Mechanismus n)



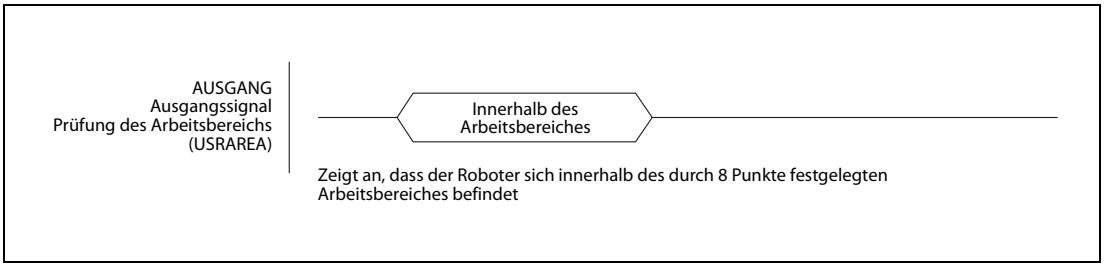
- HNDERRn (Eingang Fehler Hand n/Ausgang Fehler Hand n)



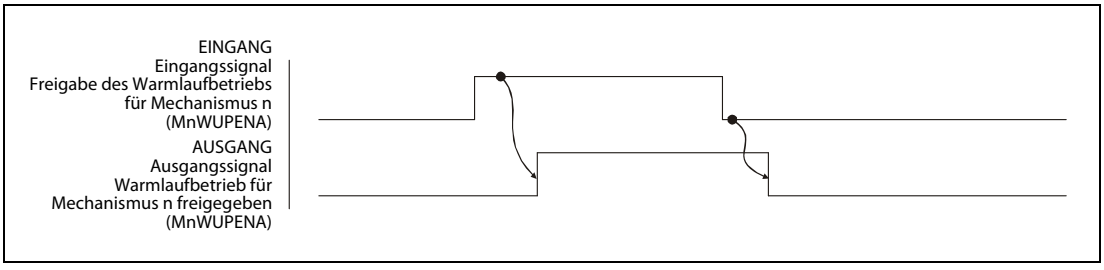
- AIRERRn (Eingang Luftdruck im Pneumatiksystem n fehlerhaft/Ausgang Pneumatikfehler im System n)



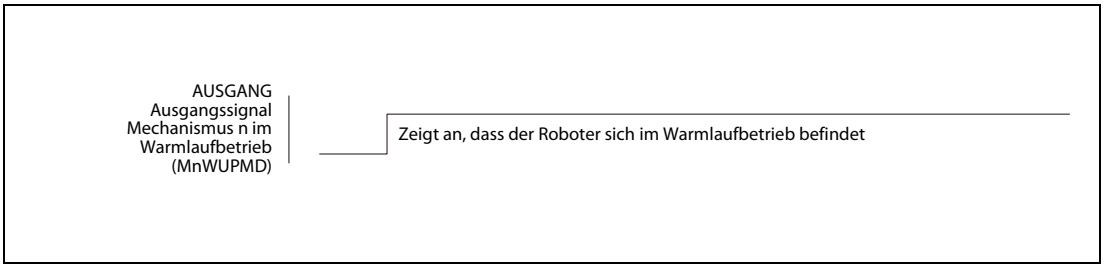
- USRAREA (Über 8 Punkte festgelegter Arbeitsbereich)



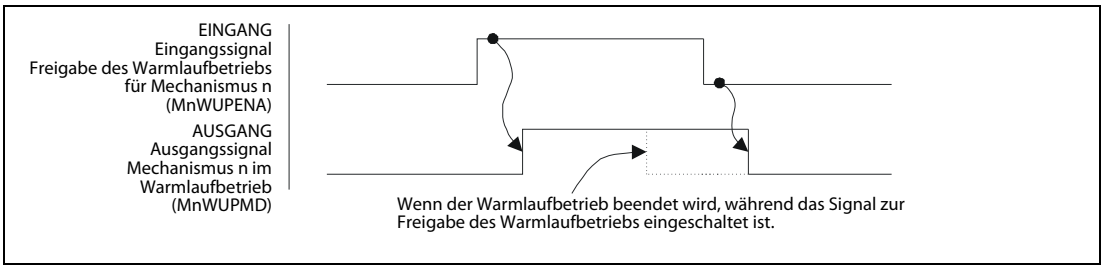
- MnWUPENA (Eingang Freigabe des Warmlaufbetriebs für Mechanismus n/Ausgang Warmlaufbetrieb für Mechanismus n freigegeben)



- MnWUPMD (Ausgangssignal Mechanismus n im Warmlaufbetrieb)



Folgende Abbildung zeigt das Zeitablaufdiagramm, wenn der Ausgang für die Signalausgabe zur Warmlaufbetriebsanzeige (MnWUPMD) und der Eingang zur Freigabe des Warmlaufbetriebs (Mn-WUPENA) gleichzeitig zugewiesen sind.



10.3.1 Programmsteuerung durch externe Signale

Zeitablaufdiagramme bei externer Steuerung

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „Programmwahl“, „Start“, „Stopp“ und „Neustart“ durch externe Signale:

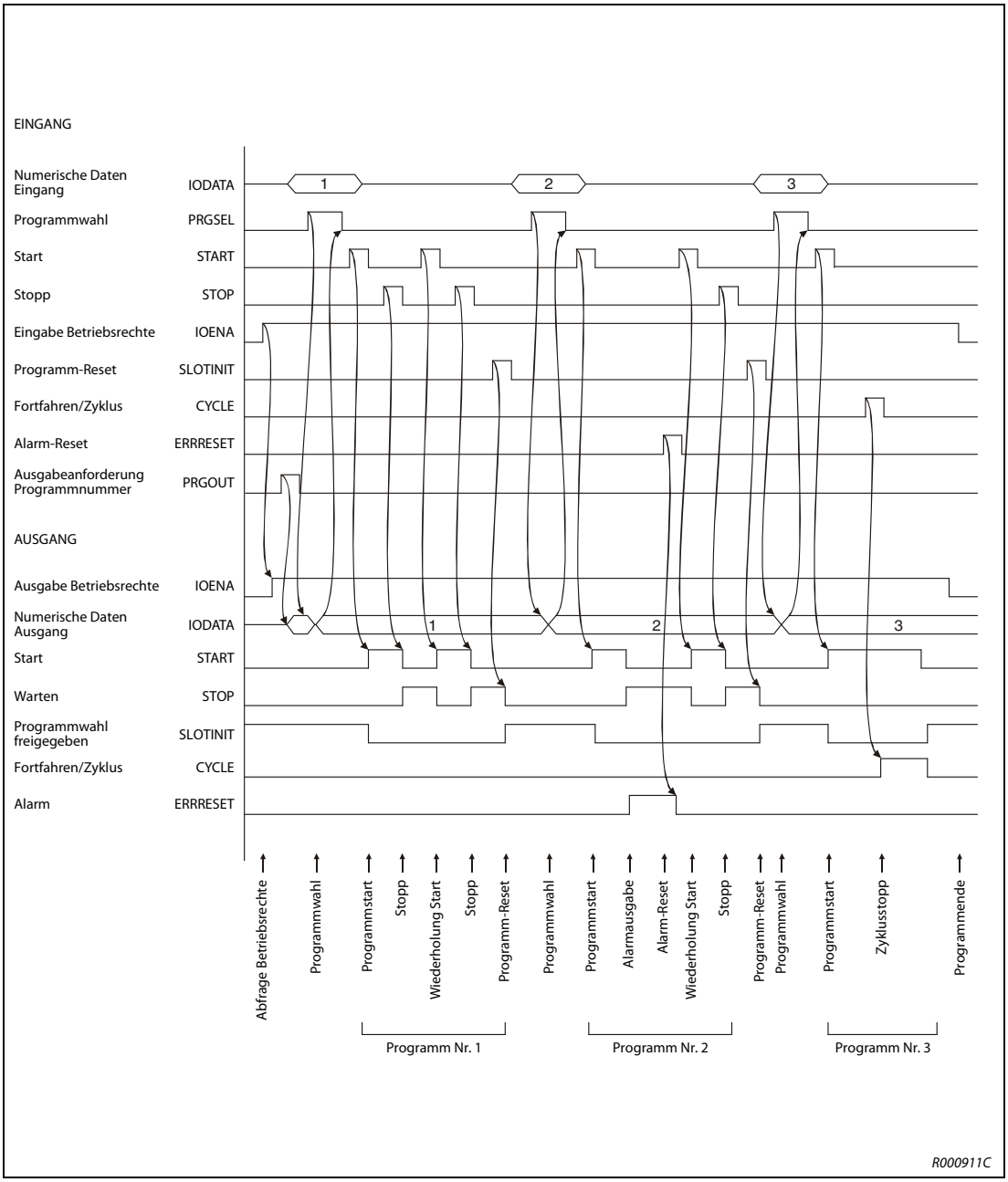


Abb. 10-3: *Zeitablaufdiagramm 1 bei externer Steuerung*

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „Servo EIN/AUS“, „Programmwahl“, „Auswahl des Geschwindigkeitsübersteuerungswertes“, „Start“, „Ausgabe der Zeilennummer“ usw. durch externe Signale:

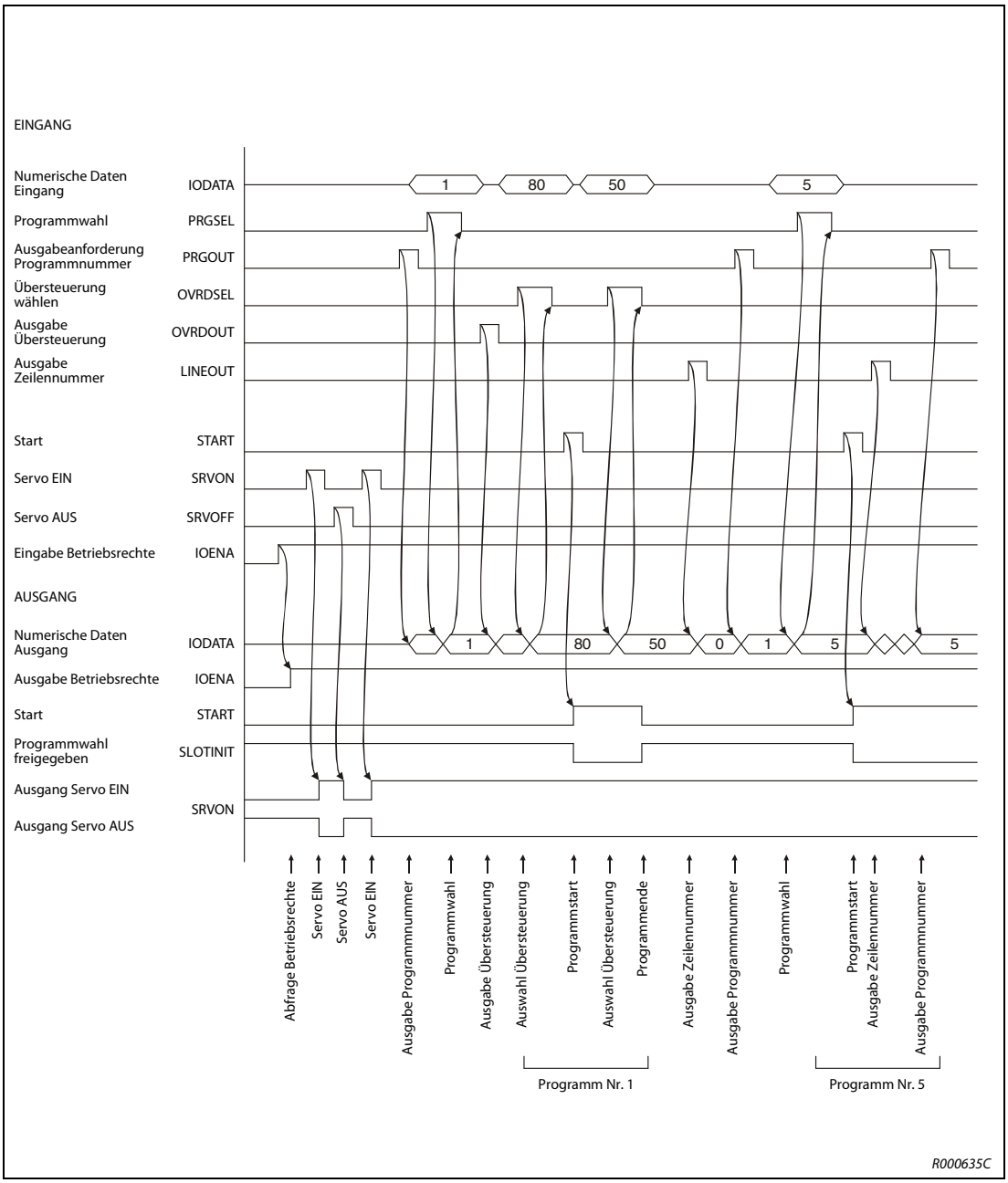


Abb. 10-4: *Zeitablaufdiagramm 2 bei externer Steuerung*

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „Fehler zurücksetzen“, „Allgemeinen Ausgang zurücksetzen“, „Programm zurücksetzen“ usw. durch externe Signale:

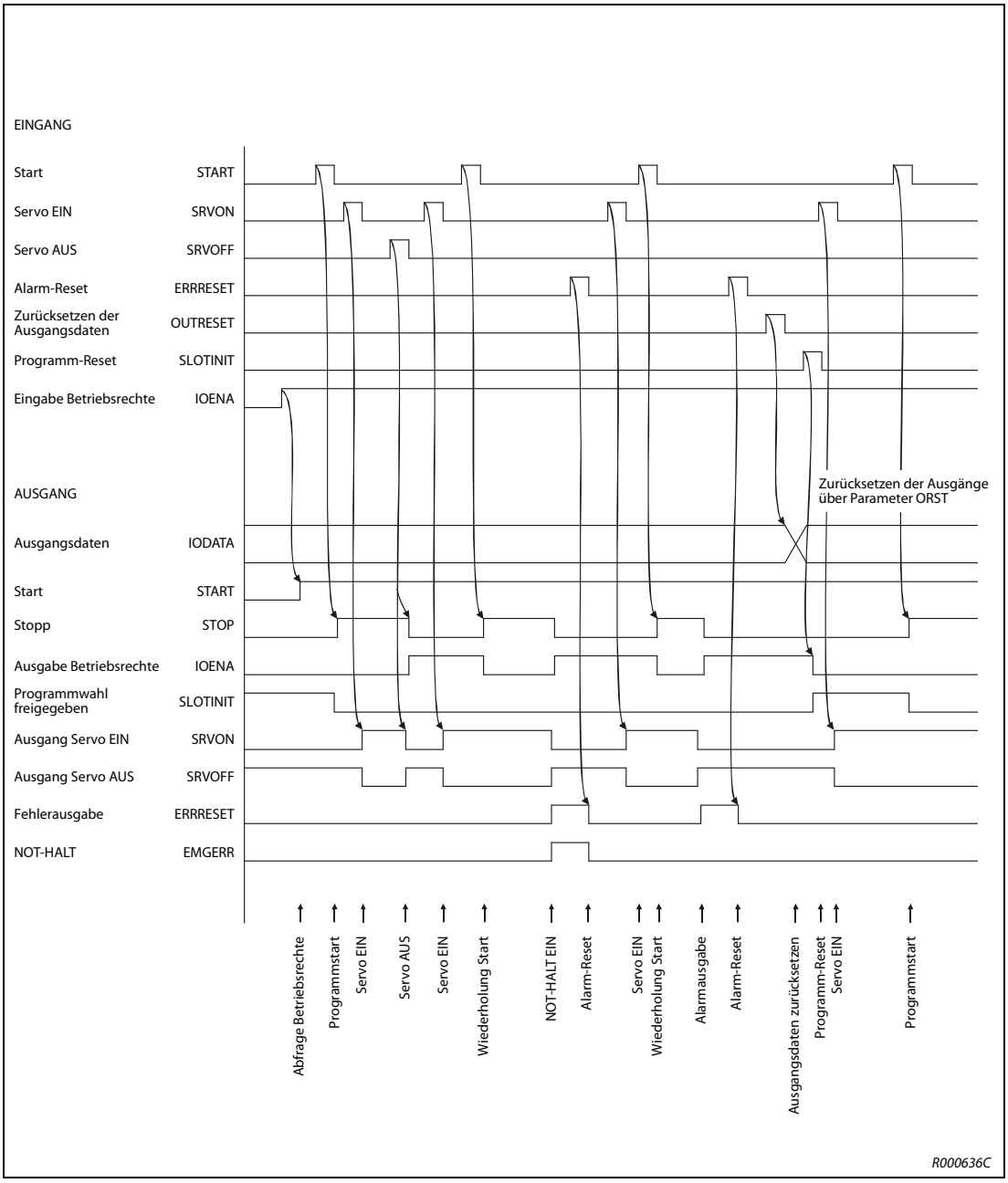


Abb. 10-5: *Zeitablaufdiagramm 3 bei externer Steuerung*

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „JOG-Betrieb“, „Anfahren der Ersatzposition“, „Programm zurücksetzen“ usw. durch externe Signale:

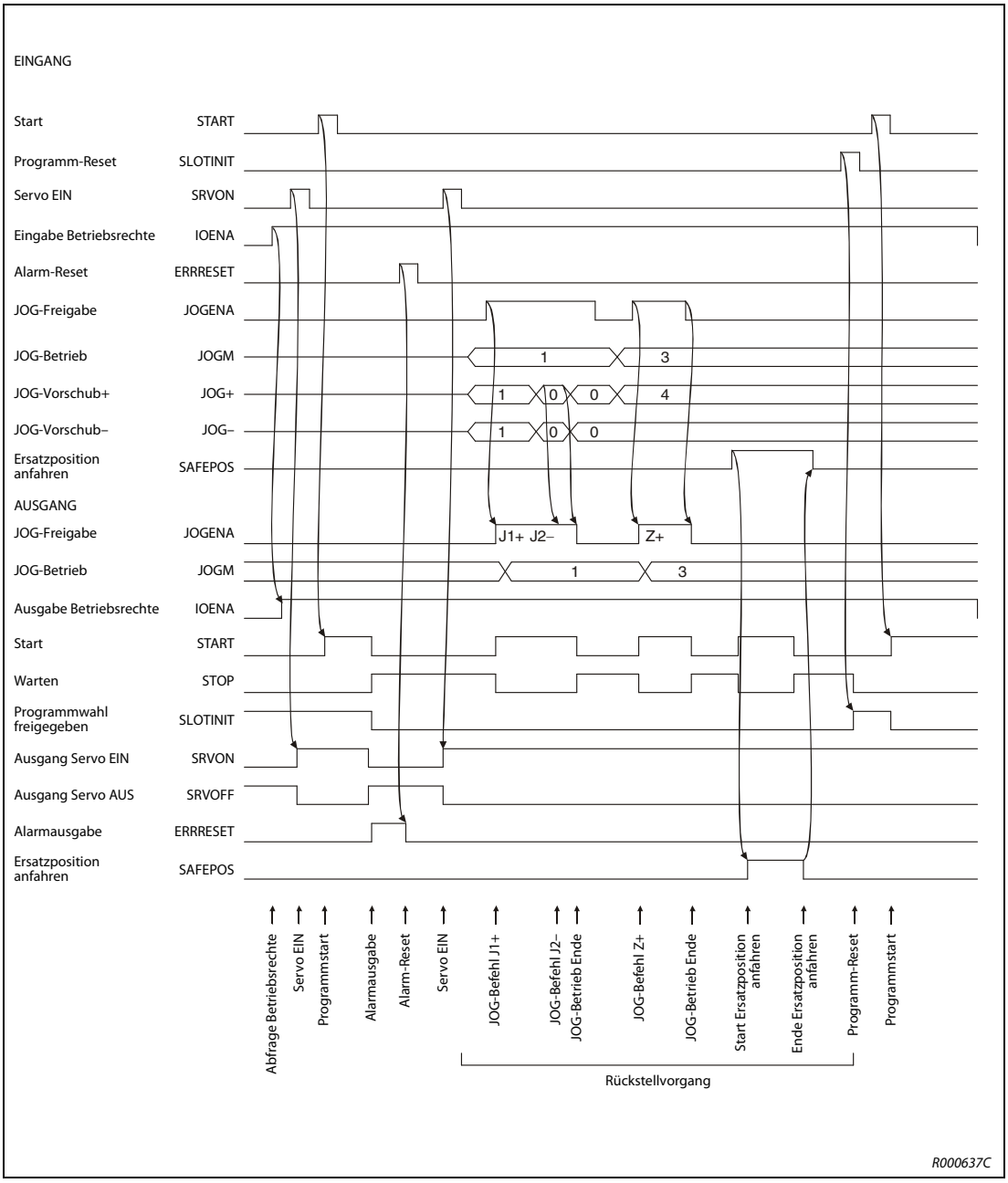


Abb. 10-6: *Zeitablaufdiagramm 4 bei externer Steuerung*

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung durch die Signale der speziellen Ein- und Ausgänge:

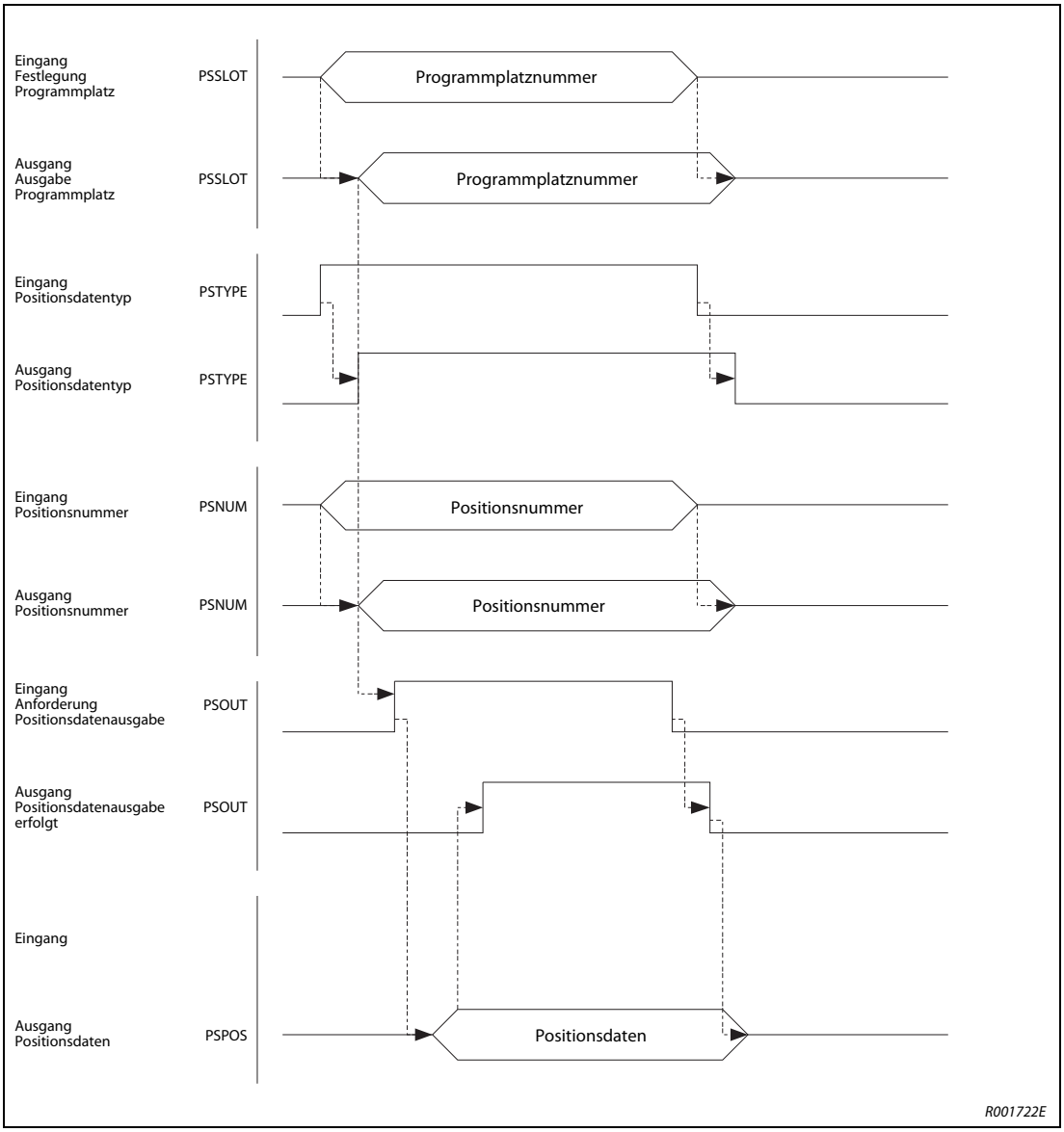


Abb. 10-7: *Zeitablaufdiagramm 5 bei externer Steuerung*

Beachten Sie die HINWEISE auf der folgenden Seite.

HINWEISE

Der Fehler 7081 wird ausgegeben, wenn die Signalnummer den zulässigen Einstellbereich des Parameters PSPOS überschreitet (32 Bits).

Der Fehler 7081 wird ausgegeben, wenn die Signalnummer den zulässigen Einstellbereich des Parameters PSSLOT überschreitet (6 Bits).

Der Fehler 7081 wird ausgegeben, wenn die Signalnummer den zulässigen Einstellbereich des Parameters PSNUM überschreitet (16 Bits).

Wird während der Anforderung zur Positionsdatenausgabe (PSOUT) der Programmplatz, der Positionsdatentyp oder die Positionsnummer geändert, dann wird der entsprechende Befehl nicht akzeptiert. Schalten Sie den Eingang zur Anforderung der Positionsdatenausgabe aus und dann wieder ein. Vergewissern Sie sich, welche Position bei der Datenausgabe betroffen ist, indem Sie die Ausgabe des Programmplatzes (PSSLOT), die Ausgabe des Positionsdatentyps (PSTYPE) und die Ausgabe der Positionsnummer (PSNUM) überprüfen.

Wenn das benötigte Programm auf den vorgesehenen Programmplatz nicht geladen ist, wird für jede Achse als Positionsdaten (PSPOS) „0x7FFFFFFF“ ausgegeben.

Wenn eine festgelegte Position nicht existiert, wird für jede Achse als Positionsdaten (PSPOS) „0x80000000“ ausgegeben.

Werden Umschaltungen vorgenommen (über die Befehle CallP, XRun oder den Parameter PRGSEL), die sich auf das ausgeführte Programm auf dem Programmplatz beziehen, wird für jede Achse als Positionsdaten (PSPOS) „0x80000000“ ausgegeben.

10.4 NOT-HALT-Eingang

Der Anschluss des NOT-HALT-Schalters erfolgt

- bei den Steuergeräten CR1D und CR2D und bei den Antriebseinheiten DU1 und DU2 über den Stecker auf der Rückseite des Geräts,
- beim Steuergerät CR3D und bei der Antriebseinheit DU3 über den Stecker des Sicherheitsmoduls.

Standardmäßig sind die NOT-HALT-Eingänge unbeschaltet (siehe Abb. 10-9). Der NOT-HALT-Schalter, der Tür-Schließkontakt und eine Freigabeeinheit müssen vom Anwender angeschlossen werden.

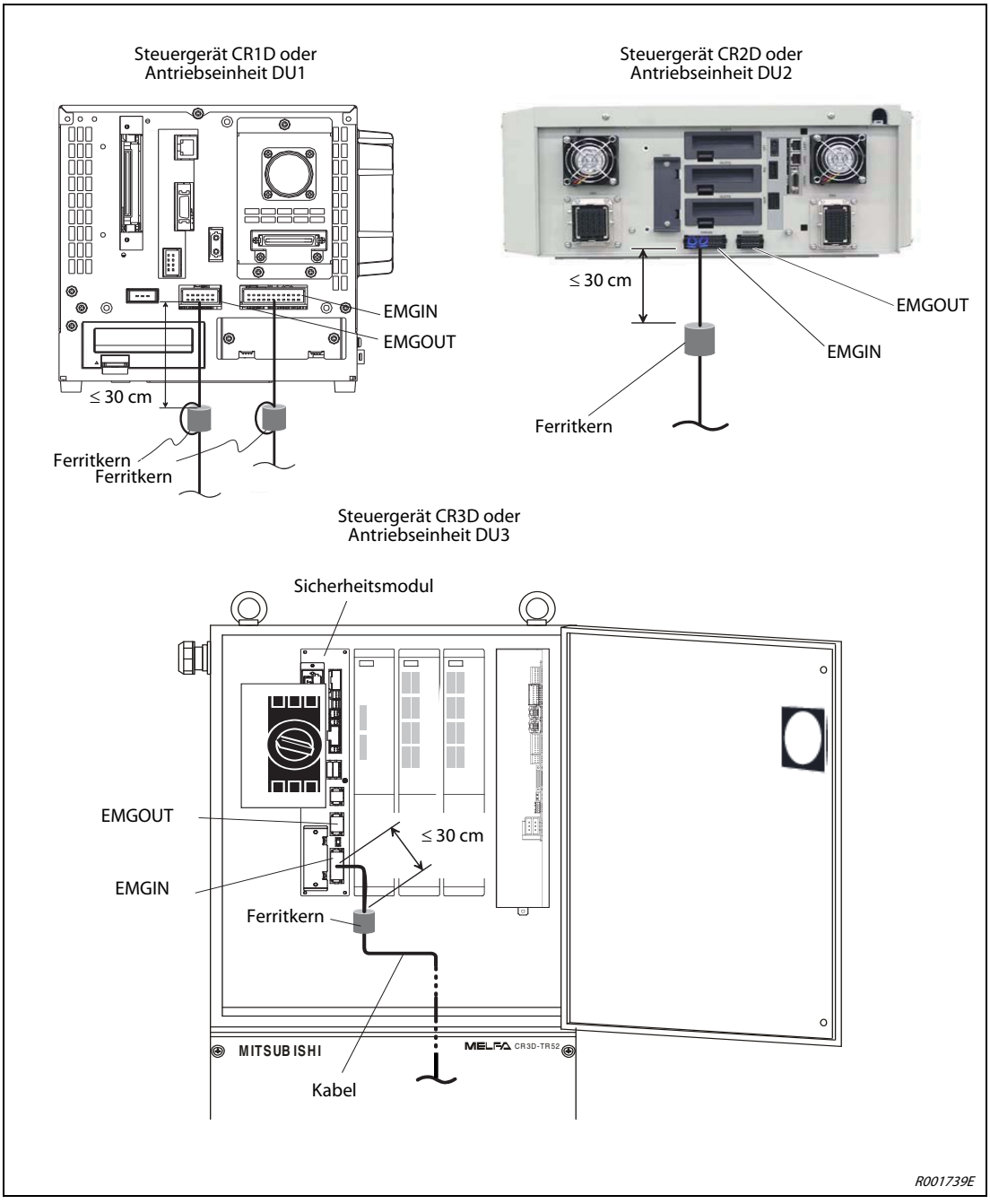


Abb. 10-8: *Anschluss der NOT-HALT-Kreise*

ACHTUNG:

Führen Sie keinen Stoßspannungstest durch.

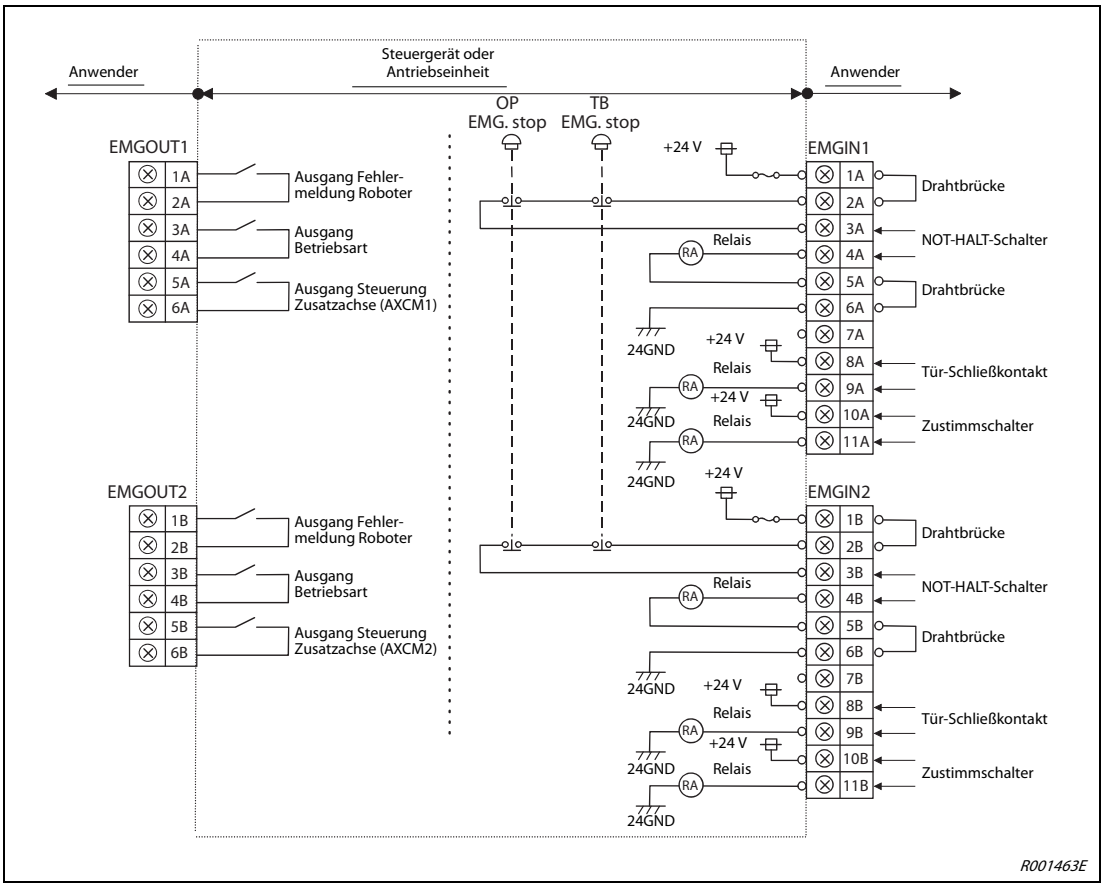


Abb. 10-9: *Interne Beschaltung der NOT-HALT-Kreise*

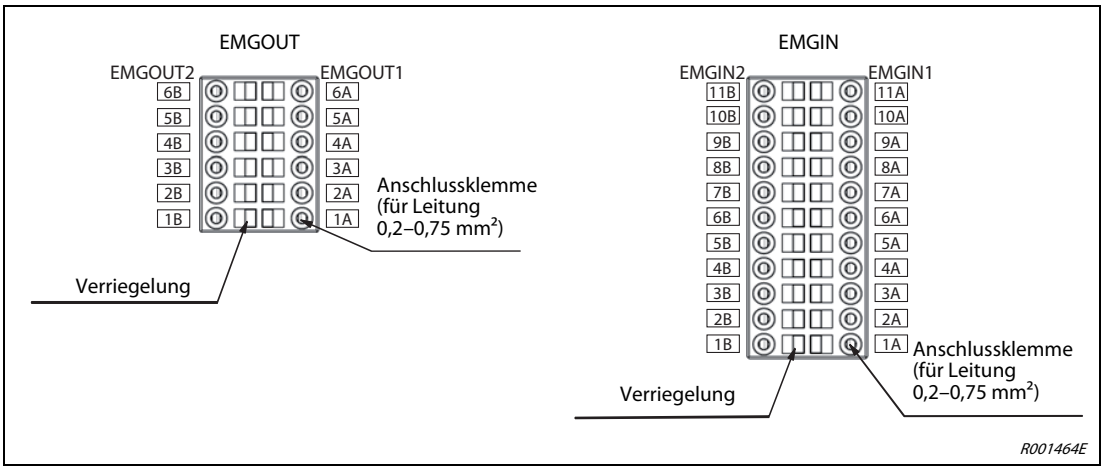


Abb. 10-10: *Anschlussbelegung der NOT-HALT-Stecker*

Anschluss an die Klemmen

① Entfernen Sie ca. 7 mm der Leitungsisolierung. Verdrillen Sie das Leitungsende vor dem Anschluss.

② Drücken Sie die Haltefeder mit einem Schraubendreher für Schlitzschrauben (Schneide: 1,4 mm bis 2,4 mm) nach unten.

③ Schließen Sie den NOT-HALT-Schalter an die Klemmen 3A-4A, 3B-4B, den Tür-Schließkontakt an die Klemmen 8A-9A, 8B-9B und den Zustimmtaster an die Klemmen 10A-11A, 10B-11B an.

④ Stecken Sie das verdrillte Leitungsende in die Anschlussklemme. Ziehen Sie den Schraubendreher aus der Verriegelung.

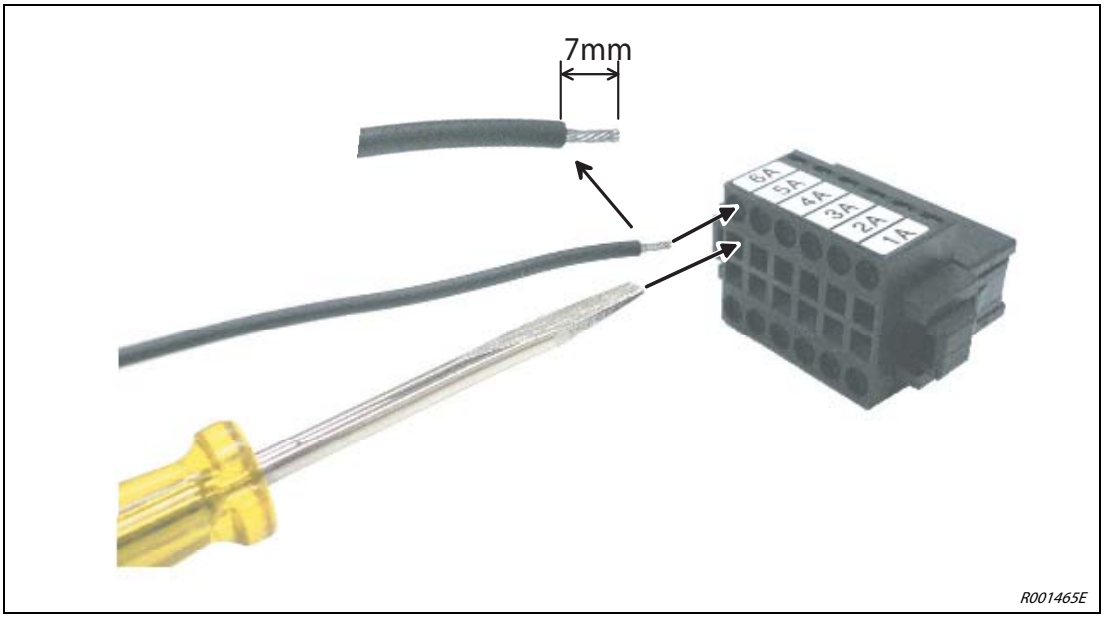


Abb. 10-11: *Anschluss an die Klemmen*

ACHTUNG:

- ***Bringen Sie den NOT-HALT-Schalter an einer leicht zugänglichen Stelle in der Nähe des Roboters an. Zeigt der Roboter ein ungewöhnliches Betriebsverhalten, ist er sofort zu stoppen.***
- ***Verdrillen Sie die Leitungen beim Anschluss an den EMGIN-Stecker sorgfältig, um Kurzschlüsse zu benachbarten Klemmen zu vermeiden. Das Ende der Leitungen darf nicht verzinnt werden, da es sich sonst während des Betriebs lösen kann.***

10.4.1 Verhalten des Roboters bei Betätigung des NOT-HALT-Schalters

Eine Betätigung des NOT-HALT-Schalters während des Roboterbetriebes führt zur Abschaltung der Servospannungsversorgung. Der weitere Verfahrweg der Handspitze und die Stoppposition sind nach der Betätigung des NOT-HALT-Schalters nicht vorhersehbar. In Abhängigkeit der Geschwindigkeit und Last kann es zum Überschwingen des Roboterarms kommen.

10.5 Anschluss eines Bediengerätes der Serie GOT1000

Das Bediengerät GT15 der GOT1000-Serie kann über die Ethernet-Schnittstelle direkt mit dem Steuergerät CRnD-700 verbunden werden. Dabei können die 256 Eingänge und 256 Ausgänge des Bediengeräts auch von dem Robotersteuergerät genutzt werden.

Weitere Einzelheiten zum Betrieb der Bediengeräte der Serie GOT1000, sowie dem grafischen Erstellungsprogramm GT Designer2 finden Sie in der jeweiligen Bedienungsanleitung.

Die Kombination eines der Bediengeräte GT15/GT16 mit den folgenden Steuergeräten mit entsprechender Software-Version erlaubt die vollständige Datenwiederherstellung der Steuergeräte auf Basis einer vorhandenen Datensicherung:

– SD-Serie: ab Software Version P8

– SQ-Serie: ab Software-Version N8

HINWEISE

Ab Software-Version S2b des Steuergerätes ist die Funktion zur direkten Verbindung mit einem GOT-Bediengerät erweitert worden. (Die Anzeige von Roboterinformationen und ein Daten-Setup sind möglich.) Weitere Informationen finden Sie im Handbuch zur Anbindung der SD-Serie an ein GOT-Bediengerät.

10.5.1 Konfigurationsbeispiel

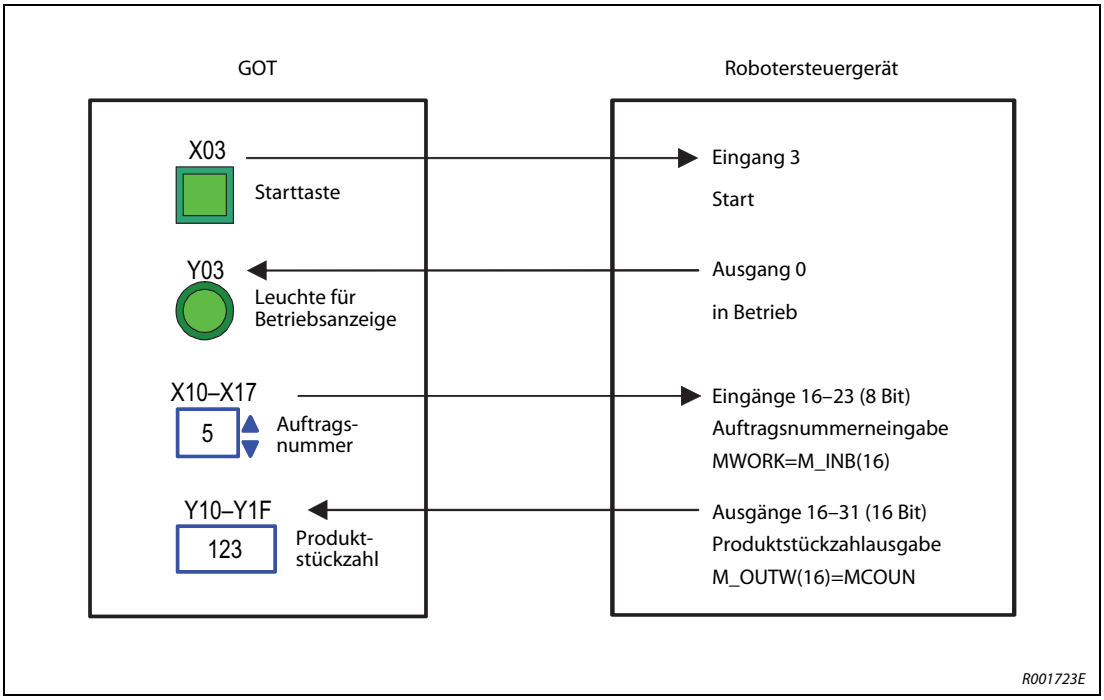


Abb. 10-12: *Beispielkonfiguration mit einem GOT-Bediengerät*

10.5.2 Technische Daten des Bediengeräts

Merkmal des GOTs	Daten	Erläuterung
Anzahl Ein/Ausgänge	256 Eingänge, 256 Ausgänge	Die Operandennamen seitens des GOTs sind X00–XFF und Y00–YFF. (Siehe auch Tab. 10-11)
Anzahl anschließbarer GOTs	1	An eine Robotersteuerung kann nur ein Bediengerät angeschlossen werden. Allerdings können an ein Bediengerät mehrere Robotesteuergeräte angeschlossen werden.
Schnittstelle	Nur Ethernet	UPD-Kommunikation Die optionale Ethernet-Schnittstelle (GT15-J71E71-100) wird für das GOT benötigt.
Verwendbare GOTs	Serie GOT1000 GT15 (in Komplettausstattung)	Nur die GT15-Bediengeräte in Komplettausstattung besitzen eine Ethernet-Schnittstelle. Daher sind andere Geräte nicht verwendbar. Die Transparent-Funktion der Computer-Support-Software (RT ToolBox2) wird nicht unterstützt (Stand Dezember 2007)
Grafisches Erstellungsprogramm	GT Designer2 ab Version 2.72A	Modell: SW2D5C-GTD2-J (japanische Version) Die englische Version ist seit Dezember 2008 verfügbar. Verwenden Sie immer die neueste Version der Software, die von der Mitsubishi-Webseite im Internet herunter geladen werden kann.

Tab. 10-10: *Daten des GOTs für das Konfigurationsbeispiel*

10.5.3 Ethernet-Anschluss an das Steuergerät

Der Anschluss des Bediengerätes an die verschiedenen Robotersteuerungen erfolgt über die Ethernet-Schnittstelle.

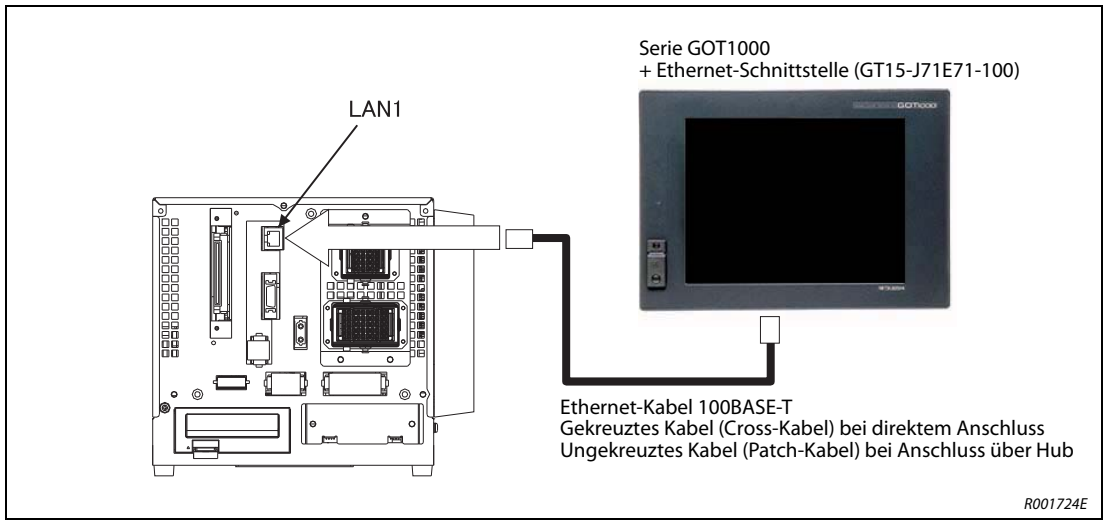


Abb. 10-13: *Modellreihe CR1n-700*

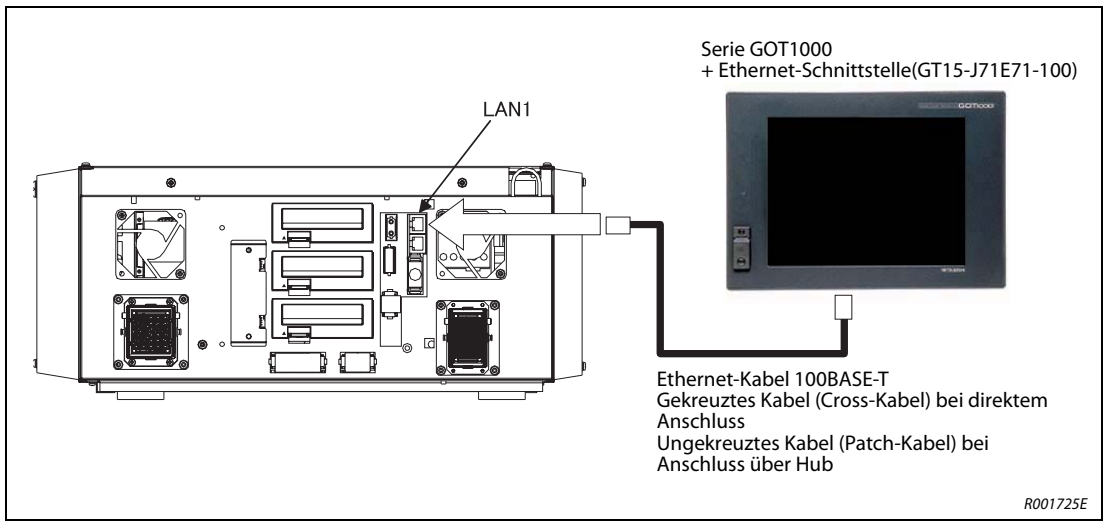


Abb. 10-14: *Modellreihe CR2n-700*

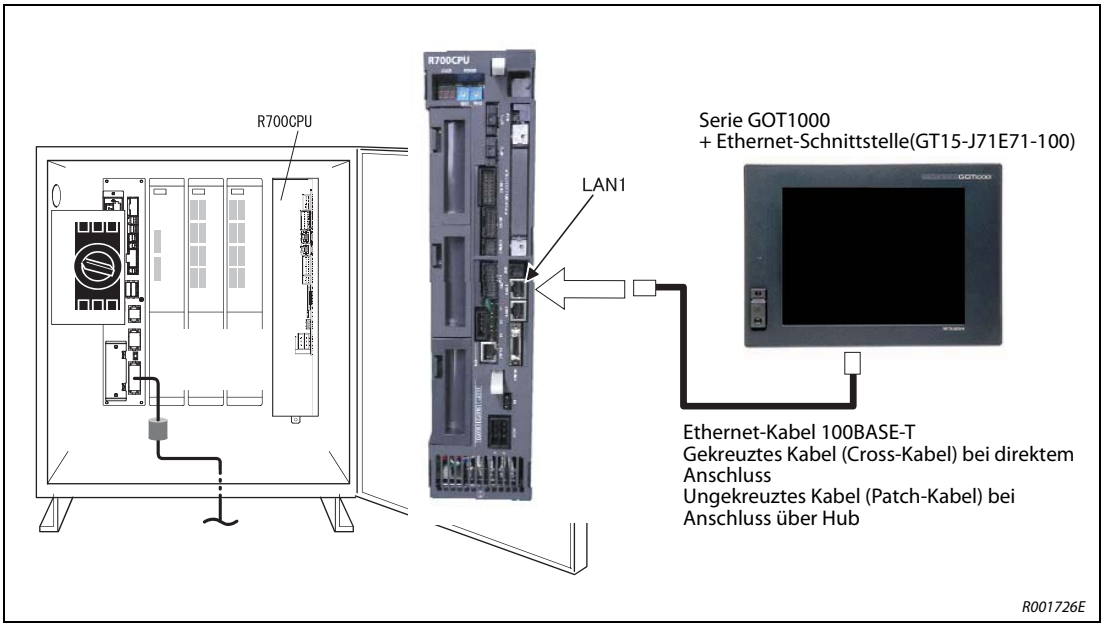


Abb. 10-15: *Modellreihe CR3n-700*

10.5.4 Einstellungen

Alle folgenden Einstellungen erfolgen mit dem grafischen Erstellungsprogramm GT Designer2.

Erstellen eines neuen Projektes

Starten Sie die Software GT Designer2 auf Ihrem Computer.

Wählen Sie im Aufklappmenü ***Project*** → ***New*** aus und befolgen Sie die angezeigten Meldungen des Assistenten bei den Einstellungen für Ihr neues Projekt. Stellen Sie die angezeigten Punkte der Bildschirmmenüs folgendermaßen ein:

① „Connection device setting“: MELSEC-QnU, Q17nD/NC/DR, CRnD-700

② „Connection I/F“: Expansion I/F-1 (first level)

③ „Communication driver“: QJ71E71/AJ71(Q)E71, Q172nNC, CRnD-700

Ethernet-Einstellungen

Wählen Sie im Bildschirmmenü ***Common settings*** → ***Ethernet*** aus. Machen Sie folgende Einstellungen:

④ „Own PC“: ?

⑤ „N/W No.“: 1

⑥ „PC No.“: 2

⑦ „Model“: CRnD-700

⑧ „IP address“ 192.168.0.20

Wenn die Werkseinstellungen der Robotersteuerung für die IP-Adresse und die GOT-Schnittstellennr. geändert wurden, prüfen Sie die Parameter NETIP und GOTPORT auf Übereinstimmung der Einstellungen.

Erstellung der Anzeigebildschirme

Erstellen Sie die Anzeigebildschirme mit der Software GT Designer 2.

HINWEISE

Netzwerkeinstellungen
Führen Sie die Einstellungen der Netzwerkgeräte mit den Einstellungen durch, wie zuvor im Abschnitt „Ethernet-Einstellungen" beschrieben.

Wenn eine einzelne Robotersteuerung an ein einzelnes Bediengerät angeschlossen wird, stellen Sie das Netzwerkgerät auf „Own PC" ein.

Wenn mehrere Robotersteuerungen an ein einzelnes Bediengerät angeschlossen werden, stellen Sie das Netzwerkgerät auf „Other PC" ein. Stellen Sie weiterhin die Netzwerknummer (N/W No.) und die Computer-Nummer (PC No.) ein. Beachten Sie, dass die CPU-Nummer nicht benötig wird und auf „0" stehen bleiben kann.

Prüfung des Imports der grafischen Erstellungsdaten in das GOT

Verbinden Sie den Computer und das Bediengerät (GOT) mit einem USB-Kabel. Stellen Sie dann im Aufklappmenü des GT Designer2 unter ***Communication*** → ***Communication with GOT*** → ***Communication setting*** auf ***USB*** ein und unter ***Project download*** → ***GOT***. Wählen Sie die korrekten Importdaten aus, importieren Sie diese in das GOT und überprüfen Sie anschließend die richtige Funktion.

Zuordnung der E/A-Signale

Die Nummern der Ein- und Ausgänge des GOT sind jeweils von 0 bis 255. Diese Nummerierung ist mit der parallelen E/A-Schnittstelle, der parallelen E/A-Einheit und dem GOT identisch.

Wenn in der Systemkonfiguration des Anwenders eine parallele E/A-Schnittstelle oder parallele E/A-Einheit vorhanden ist, ist darauf zu achten, dass die vom GOT importierten Eingangssignale an die verwendete Schnittstelle angepasst werden.

Beispiel ▽ Wenn z. B. auf Optionssteckplatz 1 der Robotersteuerung eine parallele E/A-Schnittstelle vorhanden ist, ergibt sich keine Wirkung auf die Eingänge 0 bis 31 der Robotersteuerung, auch wenn die Eingangssignale X00 bis X1F des GOT wiederholt gesetzt werden. Weiterhin beziehen sich die Eingangsnummern der parallelen E/A-Schnittstelle auf die Eingänge 0 bis 31 der Robotersteuerung.

△

Station	E/A-Nummern der Robotersteuerung	Steckplatz der parallelen E/A-Schnittstelle	Stationsnummer der parallelen E/A-Schnittstelle (Drehschaltereinstellung)	GOT1000 XY-Operanden aus Sicht des GOTs
0	0–31	Optionssteckplatz 1	0	X00-X1F / Y00-Y1F
1	32–63	Optionssteckplatz 2	1	X20-X3F / Y20-Y3F
2	64–95	Optionssteckplatz 3	2	X04-X5F / Y40-Y5F
3	96–127	—	3	X60-X7F / Y60-Y7F
4	128–159	—	4	X80-X9F / Y80-Y9F
5	160–191	—	5	XA0-XBF / YA0-YBF
6	192–223	—	6	XC0-XDF / YC0-YDF
7	224–255	—	7	XE0-XFF / YE0-YFF

Tab. 10-11: *E/A-Zuordnung Robotersteuerung – GOT*

11 Echtzeitsteuerung & Programmbeispiele

11.1 Externe Steuerung in Echtzeit

Mit jedem Taktzyklus des Steuergeräts (ca. 7,1 ms) können Positionsdaten von einer externen Quelle abgefragt werden, die direkt zur Steuerung der Roboterbewegung dienen. Zusätzlich ist es möglich, Ein- und Ausgangssignale gleichzeitig zu überwachen oder gleichzeitig Signale auszugeben.

Diese Kommunikation zur Steuerung und Überwachung in Echtzeit erfolgt mit dem MELFA-BASIC-V-Befehl Mxt.

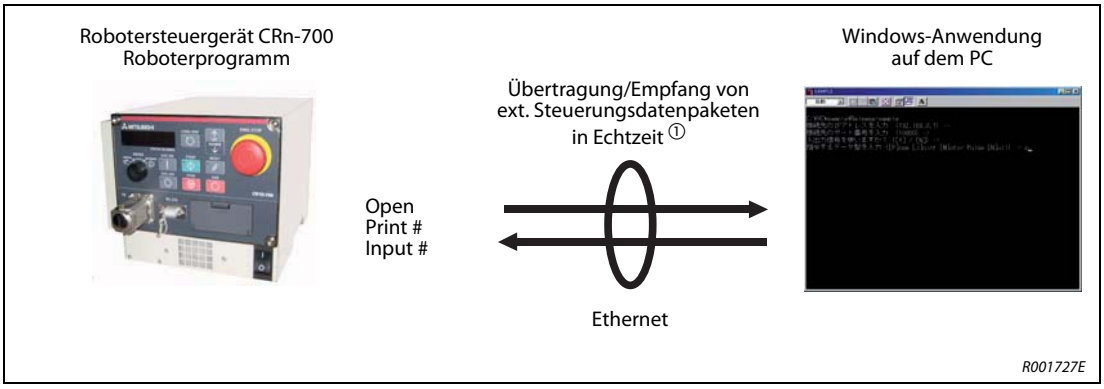


Abb. 11-1: *Echtzeitkommunikation über Ethernet*

① Abb. 11-3 zeigt das Ablaufdiagramm der externen Steuerung in Echtzeit.

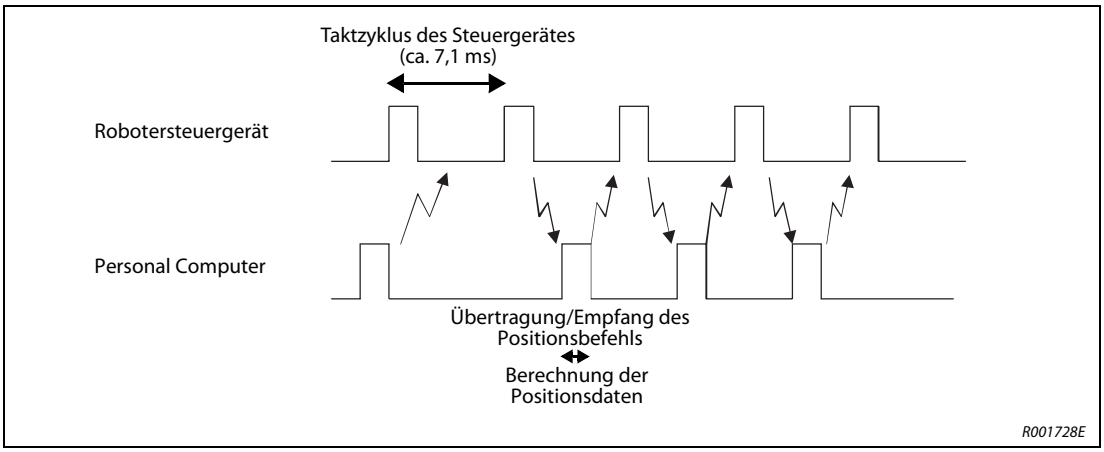


Abb. 11-2: *Zeitliche Ablauf der Echtzeitkommunikation*

Die folgende Tabelle zählt die Positionsdatentypen auf, die vom Personal Computer zum Roboter übertragen werden, um den Roboter bei jedem Taktzyklus bewegen zu können und die Datentypen, die eine Überwachung des Roboters zulassen.
Weitere Informationen zu diesem Thema finden Sie in Abschn. 6.3.56 (Befehl „Mxt (Move External)“, sowie in Abschn. 11.1.1 (Erläuterung der Kommunikationsdatenpakete).

Positionsdatentyp	Überwachungsdatentyp
[1] XYZ-Koordinatendaten	[1] XYZ-Koordinatendaten
[2] Gelenkkoordinatendaten	[2] Gelenkkoordinatendaten
[3] Motorimpulskoordinatendaten	[3] Motorimpulskoordinatendaten
	[4] XYZ-Koordinatendaten (Sollwert nach der Filterung)
	[5] Gelenkkoordinatendaten (Sollwert nach der Filterung)
	[6] Motorimpulskoordinatendaten (nach der Filterung)
	[7] XYZ-Koordinatendaten (Encoder-Rückmeldewert)
	[8] Gelenkkoordinatendaten (Encoder-Rückmeldewert)
	[9] Motorimpulskoordinatendaten (Encoder-Rückmeldewert)
	[10] Stromsollwert (%)
	[11] Stromistwert (%)

Tab. 11-1: *Übertragene Positionskommandodaten und Monitordaten über das Ethernet*

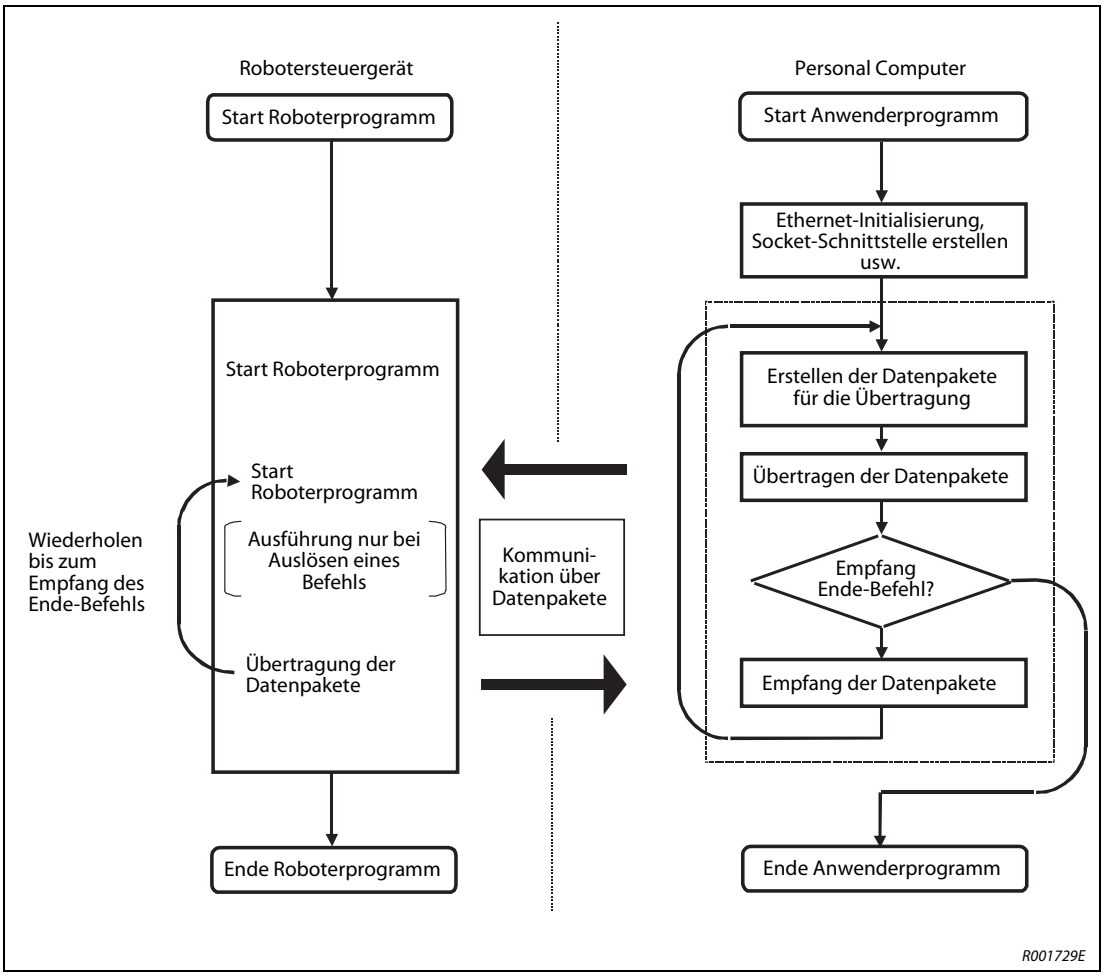


Abb. 11-3: *Ablaufdiagramm der externen Steuerung in Echtzeit*

11.1.1 Erläuterung der Kommunikation mit Datenpaketen

In diesem Abschnitt wird die Struktur der Kommunikation mit Datenpaketen bei der externen Steuerung in Echtzeit erklärt.

Für die externe Steuerung in Echtzeit werden die gleichen Datenpakete verwendet, wie für die Positionsbefehle und die Überwachung. Nur der Inhalt der Datenpakete unterscheidet sich je nach dem, ob man Daten vom Computer zur Steuerung überträgt (Befehle) oder von der Steuerung zum Computer (Überwachung).

Beachten Sie die folgenden Ausführungen zur Struktur von Datenpaketen bei der Kommunikation, sowie das „Beispielprogramm zur externen Steuerung in Echtzeit" in Abschn. 11.2.4. In der folgenden Tabelle wird der Datentyp der Programmiersprache C verwendet.

Datenart	**Datenbreite**	**Daten**
Befehl	ohne Vorzeichen, kurz (2 Byte)	Festlegung der Gültigkeit des externen Echtzeitbefehls und des Endes 0 // Externer Echtzeitbefehl ungültig 1 // Externer Echtzeitbefehl gültig 255 // Externer Echtzeitbefehl Ende
Zuordnung Übertragungsdatenart SendType	ohne Vorzeichen, kurz (2 Byte)	1) Festlegung der Art der Positionsdaten, die vom Personal Computer (Befehlsdaten) zur Steuerung übertragen werden Die erste Übertragung erfolgt ohne Daten 0 // Keine Daten 1 // XYZ-Daten 2 // Gelenkdaten 3 // Motorimpulsdaten 2) Festlegung der Art der Positionsdaten, die von der Steuerung an den Personal Computer (Überwachungsdaten) ausgegeben werden. 0 // Keine Daten 1 // XYZ-Daten 2 // Gelenkdaten 3 // Motorimpulsdaten 4 // XYZ-Daten (Position nach der Filterung) 5 // Gelenkdaten (Position nach der Filterung) 6 // Motorimpulsdaten (Position nach der Filterung) 7 // XYZ-Daten (Encoder-Rückmeldewert) 8 // Gelenkdaten (Encoder-Rückmeldewert) 9 // Motorimpulsdaten (Encoder-Rückmeldewert) 10 // Stromsollwert [%] 11 // Stromistwert [%] HINWEIS: Die Daten entsprechen denen der Datenart „RecvType", die alternativ verwendet werden kann.

Tab. 11-2: *Kommunikationsdatenpakete (1)*

Datenart	Datenbreite	Daten
Zuordnung der Antwortdaten RecvType	ohne Vorzeichen, kurz (2 Byte)	1) Festlegung der Art der Antwortdaten, die vom Personal Computer (Befehlsdaten) zur Steuerung übertragen werden. 0 // Keine Daten 1 // XYZ-Daten 2 // Gelenkdaten 3 // Motorimpulsdaten 4 // XYZ-Daten (Position nach der Filterung) 5 // Gelenkdaten (Position nach der Filterung) 6 // Motorimpulsdaten (Position nach der Filterung) 7 // XYZ-Daten (Encoder-Rückmeldewert) 8 // Gelenkdaten (Encoder-Rückmeldewert) 9 // Motorimpulsdaten (Encoder-Rückmeldewert) 10 // Stromsollwert [%] 11 // Stromistwert [%] 2) Festlegung der Art der Antwortdaten, die von der Steuerung (Überwachungsdaten) an den Personal Computer ausgegeben werden. 0 // Keine Daten 1 // XYZ-Daten 2 // Gelenkdaten 3 // Motorimpulsdaten 4 // XYZ-Daten (Position nach der Filterung) 5 // Gelenkdaten (Position nach der Filterung) 6 // Motorimpulsdaten (Position nach der Filterung) 7 // XYZ-Daten (Encoder-Rückmeldewert) 8 // Gelenkdaten (Encoder-Rückmeldewert) 9 // Motorimpulsdaten (Encoder-Rückmeldewert) 10 // Stromsollwert [%] 11 // Stromistwert [%] HINWEIS: Die Daten entsprechen denen der Datenart „SendType“ und können alternativ verwendet werden.
Reserve reserve	ohne Vorzeichen, kurz (2 Byte)	Nicht verwendet
Positionsdaten Pos / jnt / pls	POSE, JOINT oder PULSE (40 Byte) ①	1) Festlegung der Sollpositionsdaten, die vom Personal Computer (Befehlsdaten) zur Steuerung übertragen werden. Führen Sie diese Einstellung identisch zur Zuordnung der Übertragungsdatenart „SendType“ aus. 2) Festlegung der Sollpositionsdaten, die von der Steuerung an den Personal Computer (Überwachungsdaten) ausgegeben werden. POSE // XYZ-Daten [mm/rad] JOINT // Gelenkdaten [rad] PULSE // Motorimpulsdaten [Impulse] oder Stromwert [%]
Ein-/Ausgangssignalzuordnung für Datenübertragung SendIOType	ohne Vorzeichen, kurz (2 Byte)	1) Festlegung der Datenart (Ein- oder Ausgangssignal), die vom Personal Computer (Befehlsdaten) zur Steuerung übertragen wird. Stellen Sie bei Nicht-Verwendung dieser Funktion „Keine Daten“ ein. 2) Festlegung der Datenart (Ein- oder Ausgangssignal), die von der Steuerung an den Personal Computer (Überwachungsdaten) ausgegeben wird. Die Inhalte der Daten sind gleich: 0 // Keine Daten 1 // Ausgangssignal 2 // Eingangssignal
Ein-/Ausgangssignalzuordnung der Antwortdaten RecvIOType	ohne Vorzeichen, kurz (2 Byte)	1) Festlegung der Datenart (Ein- oder Ausgangssignal) bei Übertragung vom Personal Computer (Befehlsdaten) zur Steuerung, die von der Steuerung ausgegeben wird. Stellen Sie bei Nicht-Verwendung dieser Funktion „Keine Daten“ ein. 0 // Keine Daten 1 // Ausgangssignal 2 // Eingangssignal 2) Personal Computer empfängt von der Steuerung (Überwachungsdaten) Nicht verwendet

Tab. 11-2: *Kommunikationsdatenpakete (2)*

Datenart	Datenbreite	Daten

Ein-/Ausgangssignaldaten
BitTop
BitMask
IoData
	ohne Vorzeichen, kurz
	ohne Vorzeichen, kurz
	ohne Vorzeichen, kurz
	(2 Byte × 3)
		1) Festlegung der Ausgangssignaldaten, die vom Personal Computer (Befehlsdaten) zur Steuerung übertragen werden.
		2) Festlegung der Ein-/Ausgangssignaldaten, die von der Steuerung an den Personal Computer (Überwachungsdaten) ausgegeben werden.
		Die Inhalte der Daten sind gleich:
		BitTop; // Nr. des Kopf-Bits des Eingangs- oder Ausgangssignals
		BitMask; // Zuweisung der Bit-Maske (gilt nur bei Befehlsdaten)
		IoData; // Wert der Ein- oder Ausgangssignaldaten (Überwachungsdaten)
		Wert der Ausgangssignaldaten (Befehlsdaten)
		Die Datenbreite ist 16 Bit

Zählwert Zeitüberschreitung (Timeout) Tcount	ohne Vorzeichen, kurz (2 Byte)	1) Übertragung vom Personal Computer (Befehlsdaten) zur Steuerung Nicht verwendet 2) Bei Empfang von Überwachungsdaten von der Steuerung zeigt der Parameter MXTTOUT (wenn der Wert NICHT „–1“ ist) die Anzahl der Versuche an, bei denen mit der Steuerung keine Kommunikation möglich war. Wenn der Wiederholungszähler den Maximalwert erreicht, wird dieser wieder auf „0“ zurückgesetzt und beginnt von vorne. Der Zähler beginnt immer mit „0“, wenn der Befehl Mxt gegeben wird.
Zählwert Kommunikationsdaten Ccount	ohne Vorzeichen, lang (4 Byte)	1) Übertragung vom Personal Computer (Befehlsdaten) zur Steuerung Nicht verwendet 2) Bei Empfang von Überwachungsdaten von der Steuerung wird hiermit die Anzahl der Kommunikationsvorgänge angezeigt. Wenn dieser Zähler den Maximalwert erreicht, wird er wieder auf „0“ zurückgesetzt und beginnt von vorne. Der Zähler beginnt immer mit „0“, wenn der Befehl MXT gegeben wird.
Zuordnung der Antwortdaten Zusatz 1 RecvType1	ohne Vorzeichen, kurz (2 Byte)	Die Daten entsprechen denen der Datenart „RecvType“. Nicht für Anweisungen verwenden!
Reserve 1 reserve1	ohne Vorzeichen, kurz (2 Byte)	Nicht verwendet
Daten Zusatz 1 pos / jnt / pls	Einer der Positionsdaten POSE/JOINT/PULSE (40 Byte)	Die Daten entsprechen denen der Positionsdaten „pos / jnt / pls/ “. Nicht für Anweisungen verwenden!
Zuordnung der Antwortdaten Zusatz 2 RecvType2	ohne Vorzeichen, kurz (2 Byte)	Die Daten entsprechen denen der Datenart „RecvType“. Nicht für Anweisungen verwenden!
Reserve 2 reserve2	ohne Vorzeichen, kurz (2 Byte)	Nicht verwendet
Daten Zusatz 2 pos / jnt / pls	Einer der Positionsdaten POSE/JOINT/PULSE (40 Byte)	Die Daten entsprechen denen der Positionsdaten „pos / jnt / pls/ “. Nicht für Anweisungen verwenden!
Zuordnung der Antwortdaten Zusatz 3 RecvType3	ohne Vorzeichen, kurz (2 Byte)	Die Daten entsprechen denen der Datenart „RecvType“. Nicht für Anweisungen verwenden!
Reserve 3 reserve3	ohne Vorzeichen, kurz (2 Byte)	Nicht verwendet
Daten Zusatz 3 pos / jnt / pls	Einer der Positionsdaten POSE/JOINT/PULSE (40 Byte)	Die Daten entsprechen denen der Positionsdaten „pos / jnt / pls/ “. Nicht für Anweisungen verwenden!

Tab. 11-2: *Kommunikationsdatenpakete (3)*

① Beachten Sie auch das Beispielprogramm zur Definition der Header-Datei „strdef.h“ in Abschn. 11.2.4 in Bezug auf die Paketdatenstrukturen bei der Kommunikation.

11.2 Beispielprogramme

Im Anschluss folgen die Beispielprogramme für die Ethernet-Schnittstelle.

11.2.1 Beispielprogramm der Datenverbindung (Daten-Link)

Das Programm für die Datenverbindung zu Microsoft Visual Basic 5.0/6.0 (im weiteren Verlauf mit VB bezeichnet) wird nachfolgend beschrieben.

Nachfolgend in Kürze der Ablauf der Programmerstellung. Der Betrieb und die Anwendung von VB ist in der zum Programm gehörenden Bedienungsanleitung beschrieben.

① Vorbereitung für „Winsock Control“

② Erzeugung des Programmbildschirms

③ Programm (Form1.frm)
Das Programm ist in zwei Teile unterteilt. Verwenden Sie das zu Ihrem System passende Programm:

– Programm für einen Client
(Der Personal Computer arbeitet als Client und das Steuergerät als Server.)

– Programm für einen Server
(Der Personal Computer arbeitet als Server und das Steuergerät als Client.)

Die Ablaufschritte ① und ② sind für den Client- und Serverbetrieb identisch.

Für die Erstellung dieses Programms ist VB als „Professional Edition“ oder als „Enterprise Edition“ notwendig. Die Lernversion (Learning Edition) von VB ist hierfür nicht geeignet, da die Steuerung der Windows-Socket hiermit nicht möglich ist.

① Vorbereitung für „Winsock Control“
In diesem Schritt wird dem Projekt die „Winsock Control“ zugefügt.

Starten Sie VB durch Doppelklicken auf die betreffende EXE-Datei.
Nach Klicken auf ***component*** im Menü ***project*** erscheint der folgende Auswahlbildschirm. Auf der Registerkarte ***Control*** muss das Kontrollkästchen für „Microsoft Winsock Control ☐☐“ aktiviert werden (☐☐ bezeichnet die jeweilige Version).

Danach erscheint im unteren rechten Teil der Werkzeugleiste (in der Abbildung rechts) die Anwendung „Winsock“.

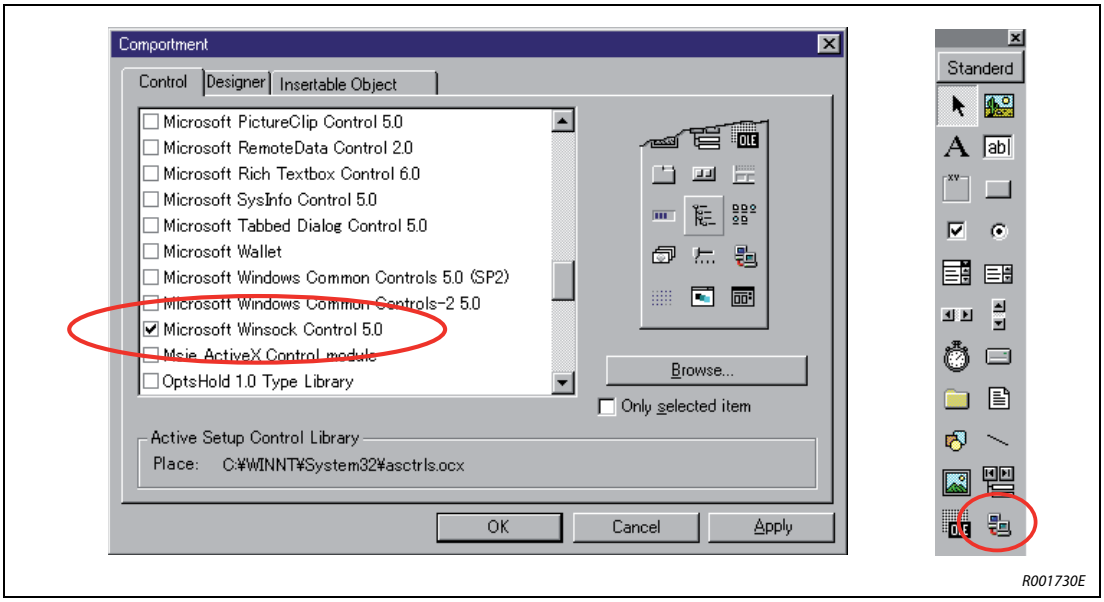


Abb. 11-4: *Menübildschirm „Comportment“*

② Erzeugung des Programmbildschirms
Auf dem Programmbildschirm sind vier Textfelder, ein Betätigungsfeld, ein Kontrollkästchen und ein Winsock-Steuerfeld angeordnet.
Die hauptsächlichen Änderungspunkte sind nachfolgend dargestellt:

Objektbezeichnung	**Eigenschaften**	**Einstellung**
Form1	Caption	Data link
Command1	Caption	Send
	Enabled	False
Text1	Text	192.168.0.1
Text2	Text	10003
Text3	Text	
Text4	MultiLine	True
	ScrollBars	2-Vertical
Check1	Caption	Connection

Tab. 11-3: *Einstellung der Eigenschaften des Programmbildschirms*

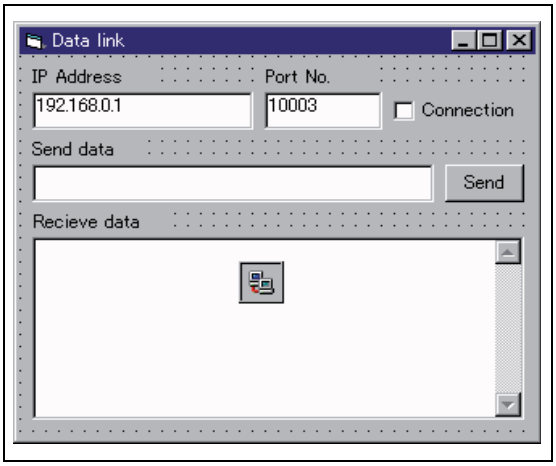


Abb. 11-5:
Programmbildschirm „Data link“

③ Programm (Form1.frm)

```
VERSION 5.00
Object = "{248DD890-BB45-11CF-9ABC-0080C7E7B78D}#1.0#0"; "MSWINSCK.OCX"

Begin VB.Form Form1                         'Beginn der Bildschirmeinstellungen
   Caption         =  "Data link"
   ClientHeight    =  3795
   ClientLeft      =  60
   ClientTop       =  345
   ClientWidth     =  4800
   LinkTopic       =  "Form1"
   ScaleHeight     =  3795
   ScaleWidth      =  4800
   StartUpPosition =  3'                    'vordefinierter Windows-Wert
   Begin MSWinsockLib.Winsock Winsock1
      Left         =  2040
      Top          =  2040
      _ExtentX     =  741
      _ExtentY     =  741
   End
   Begin VB.CommandButton Command1
      Caption      =  "Send"
      Enabled      =  0                     'deaktiviert
      Height       =  375
      Left         =  3960
      TabIndex     =  6
      Top          =  1080
      Width        =  735
   End
   Begin VB.CheckBox Check1
      Caption      =  "Connection"
      Height       =  375
      Left         =  3960
      TabIndex     =  4
      Top          =  360
      Width        =  735
   End
   Begin VB.TextBox Text4
      Height       =  815
      Left         =  120
      MultiLine    =  -1                    'richtig
      ScrollBars   =  2                     'vertikal
      TabIndex     =  7
      Top          =  1800
      Width        =  4575
   End
   Begin VB.TextBox Text3
      Height       =  375
      Left         =  120
      TabIndex     =  5
      Top          =  1080
      Width        =  3735
   End
   Begin VB.TextBox Text2
      Height       =  375
      Left         =  2280
      TabIndex     =  3
      Text         =  "10003"
      Top          =  360
      Width        =  1575
   End
```

```
    Begin VB.TextBox Text1
        Height          =   375
        Left            =   120
        TabIndex        =   2
        Text            =   "192.168.0.1"
        Top             =   360
        Width           =   2055
    End
    Begin VB.Label Label4
        Caption         =   "Receive data"
        Height          =   195
        Left            =   120
        TabIndex        =   9
        Top             =   1560
        Width           =   975
    End
    Begin VB.Label Label3
        Caption         =   "Send data"
        Height          =   195
        Left            =   120
        TabIndex        =   8
        Top             =   840
        Width           =   975
    End
    Begin VB.Label Label2
        Caption         =   "Port No."
        Height          =   195
        Left            =   2280
        TabIndex        =   1
        Top             =   120
        Width           =   975
    End
    Begin VB.Label Label1
        Caption         =   "IP address"
        Height          =   255
        Left            =   120
        TabIndex        =   0
        Top             =   120
        Width           =   1095
    End
End                             'Ende der Bildschirmeinstellungen

Attribute VB_Name               =   "Form1"
Attribute VB_GlobalNameSpace    =   False
Attribute VB_Creatable          =   False
Attribute VB_PredeclaredId      =   True
Attribute VB_Exposed            =   False
```

11.2.2 Programm für einen Client

Der Personal Computer arbeitet als Client und das Steuergerät als Server.

```
Option Explicit
Dim RecvData() As Byte

Private Sub Check1_Click()          'Ausführen bei Aktivierung des Kontrollkästchens „Connection“
If Check1.Value Then
    Winsock1.RemoteHost= Text1.Text
    Winsock1.RemotePort= Text2.Text
    Winsock1.Connect
Else
    Winsock1.Close
End If
End Sub

Private Sub Winsock1_Connect()      'Ausführen, wenn Netzwerkverbindung möglich
    Command1.Enabled = True
End Sub

Private Sub Winsock1_Close()        'Ausführen, wenn Netzwerk geschlossen ist
    Check1.Value = False
End Sub

Private Sub Command1_Click()        'Ausführen bei Anklicken der Taste „Transmission“
    Winsock1.SendData (Text3.Text)
End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)      'Ausführen bei Datenempfang
    If bytesTotal > 0 Then
        ReDim RecvData(bytesTotal - 1)
        Call Winsock1.GetData(RecvData, , bytesTotal)
        Text4.SelStart = Len(Text4.Text)
        Text4.SelText = StrConv(RecvData, vbUnicode)
    End If
End Sub

Private Sub Winsock1_Error(ByVal Number As Integer, _
    Description As String, ByVal Scode As Long, _
    ByVal Source As String, ByVal HelpFile As String, _
    ByVal HelpContext As Long, CancelDisplay As Boolean)      'Ausführen bei Fehler im
                                                               Window-Socket
    Check1.Value = False
    Command1.Enabled = False
    Winsock1.Close
    MsgBox " Error:" & Number & "(" & Description & ")"
End Sub
```

11.2.3 Programm für einen Server

Der Personal Computer arbeitet als Server und das Steuergerät als Client.

```
Option Explicit
Dim RecvData() As Byte

Private Sub Form_Load()
    Text1.Enabled = False              'Editieren der IP-Adresse unterbinden.
End Sub

Private Sub Check1_Click()          'Ausführen bei Aktivierung des Kontrollkästchens „Connection“
    If Check1.Value Then
        Text1.Text = Winsock1.LocalIP
        Winsock1.LocalPort = Text2.Text
        Winsock1.Listen
    Else
        Command1.Enabled = False
        Winsock1.Close
    End If
End Sub

Private Sub Winsock1_Connect()      'Ausführen, wenn Netzwerkverbindung möglich
    Command1.Enabled = True
End Sub

Private Sub Winsock1_Close(          )'Ausführen, wenn Netzwerk geschlossen ist
    Check1.Value = False
End Sub

Private Sub Command1_Click()        'Ausführen bei Anklicken der Taste „Transmission“
    Winsock1.SendData (Text3.Text)
End Sub

Private Sub Winsock1_ConnectionRequest(ByVal requestID As Long)    'Ausführen bei Anforde-
                                                                   rung einer Verbindung
    If Winsock1.State <> sckClosed Then Winsock1.Close
    Winsock1.Accept requestID
    Command1.Enabled = True
End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)      'Ausführen bei Datenempfang
    If bytesTotal > 0 Then
        ReDim RecvData(bytesTotal - 1)
        Call Winsock1.GetData(RecvData, , bytesTotal)
        Text4.SelStart = Len(Text4.Text)
        Text4.SelText = StrConv(RecvData, vbUnicode)
        Text4.Text = Text4.Text & vbCrLf
    End If
End Sub

Private Sub Winsock1_Error(ByVal Number As Integer, _
    Description As String, ByVal Scode As Long, _
    ByVal Source As String, ByVal HelpFile As String, _
    ByVal HelpContext As Long, CancelDisplay As Boolean)  'Ausführen bei Fehler im Window-Socket
    Check1.Value = False
    Command1.Enabled = False
    Winsock1.Close
    MsgBox "Error:" & Number & "(" & Description & ")"
End Sub
```

Beziehung zwischen der Dateinummer COMn: beim Öffnen einer Kommunikationsleitung und dem Parameter COMDEV (1), (2), (3), (4), (5), (6), (7), (8)

Datei Nr. der Kommunikationsleitung	COMDEV
COM1:	(1)
COM2:	(2)
COM3:	(3)
COM4:	(4)
COM5:	(5)
COM6:	(6)
COM7:	(7)
COM8:	(8)

Gerätezuordnung über Namen (OPT11 bis OPT19) mit dem Parameter COMDEV und Einstellung des Protokolls auf Data-link

Schnittstellennr.	Gerätename	Protokoll	
	COMDEV Einstellung	CPRCE	Einstellung
10001	OPT11	CPRCE11	2
10002	OPT12	CPRCE12	2
10003	OPT13	CPRCE13	2
10004	OPT14	CPRCE14	2
10005	OPT15	CPRCE15	2
10006	OPT16	CPRCE16	2
10007	OPT17	CPRCE17	2
10008	OPT18	CPRCE18	2
10009	OPT19	CPRCE19	2

11.2.4 Beispielprogramm für externe Echtzeitsteuerung

Nachfolgend wird ein Programm gezeigt, welches die Datenverbindung (Daten-Link) über Microsoft Visual C++5.0/6.9 (im weiteren Verlauf als VC bezeichnet) einrichtet.

Nachfolgend in Kürze der Ablauf der Programmerstellung. Der Betrieb und die Anwendung von VC ist in der zum Programm gehörenden Bedienungsanleitung beschrieben.

① Neues Projekt erstellen

② Das Programm „sample.cpp/strdef.h“ erstellen

Erläuterung der Schritte ① und ②

① Neues Projekt erstellen

Starten Sie VC und erzeugen Sie ein neues Projekt.
Stellen Sie den Namen auf „Win32 Console Application“ ein.

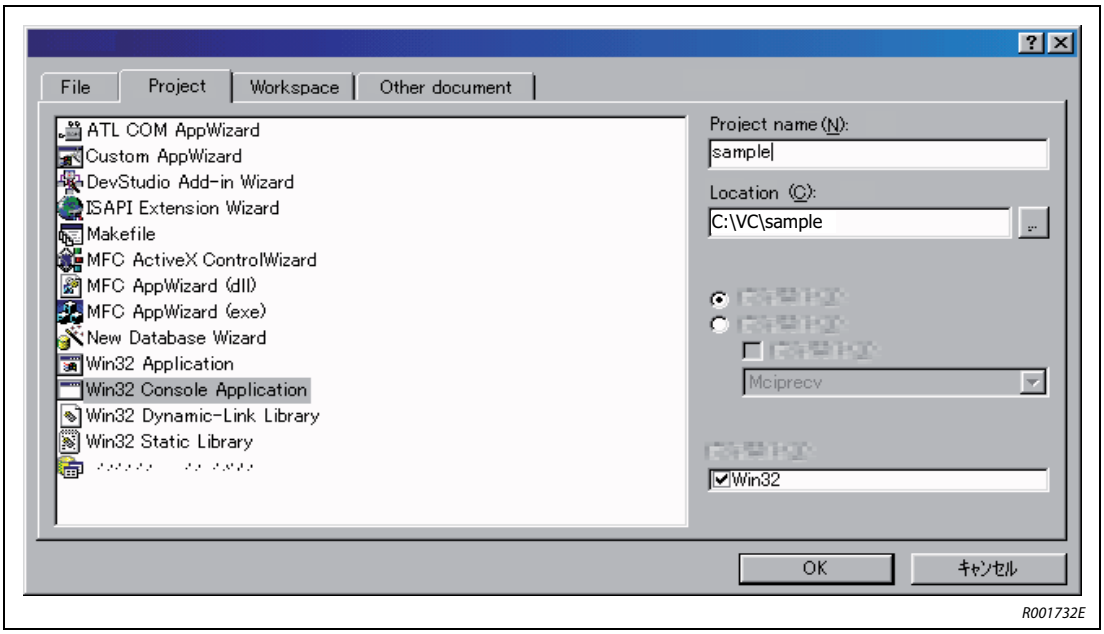


Abb. 11-6: *Zufügen der wsock32.lib zum object/library-Modul mit Hilfe der Projekteinstellung*

② Das Programm „sample.cpp/strdef.h“ erstellen

Hierzu muss die Header-Datei strdef.h und die Quelldatei sample.cpp neu erstellt werden.

HINWEIS

für das Kompilieren
Verwenden Sie die Anfangseinstellung von 8 Bytes bei der Alignment-Compiler-Option des Strukturmembers. Nach Erstellen eines neuen Projektes in Visual C++ mit der Anfangseinstellung gibt es kein Problem. Zu weiteren Informationen dazu beachten Sie auch die Hilfe von C++.

Header-Datei strdef.h

```
//*****************************************************************************************
// Beispielprogramm für Echtzeitsteuerung
// Definition der Header-Datei für die Paketdatenstruktur für die Kommunikation
//*****************************************************************************************
// strdef.h
/*************************************************************************/
/*Gelenkkoordinatensystem (Nicht verwendete Achsen auf 0 setzen)*/
/*Beachten Sie die mit jedem Roboter gelieferte Bedienungsanleitung*/
/*zu den Einzelheiten jedes einzelnen Elements. */
/*************************************************************************/

typedef struct{
    float       j1;// Winkel J1-Achse (radiant)
    float       j2;// Winkel J2-Achse (radiant)
    float       j3;// Winkel J3-Achse (radiant)
    float       j4;// Winkel J4-Achse (radiant)
    float       j5;// Winkel J5-Achse (radiant)
    float       j6;// Winkel J6-Achse (radiant)
    float       j7;// Zusatzachse 1 (Winkel J7-Achse) (radiant)
    float       j8;// Zusatzachse 2 (Winkel J8-Achse) (radiant)
} JOINT;

/*************************************************************************/
/*XYZ-Koordinatensystem (Nicht verwendete Achsen auf 0 setzen)*/
/*Beachten Sie die mit jedem Roboter gelieferte Bedienungsanleitung*/
/*zu den Einzelheiten jedes einzelnen Elements. */

/*************************************************************************/

typedef struct{
    float       x;// Koordinatenwert X-Achse (mm)
    float       y;// Koordinatenwert Y-Achse (mm)
    float       z;// Koordinatenwert Z-Achse (mm)
    float       a;// Koordinatenwert A-Achse (radiant)
    float       b;// Koordinatenwert B-Achse (radiant)
    float       c;// Koordinatenwert C-Achse (radiant)
    float       l1;// Zusatzachse 1 (mm oder radiant)
    float       l2;// Zusatzachse 2 (mm oder radiant)
} WORLD;

typedef struct{
    WORLD w;
    unsigned int sflg1;//Stellungsmerker 1
    unsigned int sflg2;//Stellungsmerker 2
} POSE;

/*************************************************************************/
/*Impulskoordinatensystem (Nicht verwendete Achsen auf 0 setzen)*/
/*Diese Koordinaten geben für jede Achse*/
/*einen Motorimpulswert an. */
/*************************************************************************/
typedef struct{
    long p1;// Achsenmotor 1
    long p2;// Achsenmotor 2
    long p3;// Achsenmotor 3
    long p4;// Achsenmotor 4
    long p5;// Achsenmotor 5
    long p6;// Achsenmotor 6
    long p7;// Zusatzachse 1 (Achsenmotor 7)
    long p8;// Zusatzachse 2 (Achsenmotor 8)

} PULSE;
```

```
/************************************************************/
/*Kommunikationspaketdaten der Echtzeitfunktion */
/************************************************************/
typedef struct enet_rtcmd_str {
    unsigned short Command;//Befehl
#define MXT_CMD_NULL0//Externer Echtzeit-Befehl ungültig
#define MXT_CMD_MOVE1// Externer Echtzeit-Befehl gültig
#define MXT_CMD_END255//Ende externer Echtzeit-Befehl

    unsigned short SendType;//Zuordnung Befehlsdatentyp
    unsigned short RecvType;//Zuordnung Überwachungsdatentyp

////////////// Befehls- oder Überwachungsdatentyp ///
#define MXT_TYP_NULL0//Keine Daten
//Für Befehl und Überwachung//////////////////
#define MXT_TYP_POSE1//XYZ-Daten
#define MXT_TYP_JOINT2//Gelenkdaten
#define MXT_TYP_PULSE3 //Impulsdaten
////////////// Für Positionsüberwachung ///
#define MXT_TYP_FPOSE4// XYZ-Daten (nach der Filterung)
#define MXT_TYP_FJOINT5// Gelenkdaten (nach der Filterung)
#define MXT_TYP_FPULSE6// Impulsedaten (nach der Filterung)
#define MXT_TYP_FB_POSE7// XYZ-Daten (Encoder-Rückmeldewert)
#define MXT_TYP_FB_JOINT8// Gelenkdaten (Encoder-Rückmeldewert)
#define MXT_TYP_FB_PULSE9// Impulsdaten (Encoder-Rückmeldewert)
//Für Stromüberwachung ///////////////////
#define MXT_TYP_CMDCUR10//Stromsollwert
#define MXT_TYP_FBKCUR11//Stromistwert

    unsigned short reserve;// Reserve
    union rtdata {//Befehlsdaten
        POSE pos;//Typ XYZ [mm/rad]
        JOINT jnt;//Typ Gelenk [rad]
        PULSE pls;//Typ Impuls [pls]
long lng1[8];//Typ Ganze Zahl [% / non-unit]
    } dat;

    unsigned short SendIOType;// Ein-/Ausgangssignalzuordnung für Datenübertragung
    unsigned short RecvIOType;// Ein-/Ausgangssignalzuordnung der Antwortdaten
#define MXT_IO_NULL0//Keine Daten
#define MXT_IO_OUT1//Ausgangssignal
#define MXT_IO_IN2//Eingangssignal

    unsigned short BitTop;// Nr. Kopf-Bit
    unsigned short BitMask;// Zuordnung der Bit-Maske (0x0001-0xffff)
    unsigned short IoData;// Ein-/Ausgangssignaldaten (0x0000-0xffff)

    unsigned short TCount;// Zählwert Zeitüberschreitung
    unsigned long CCount;// Zählwert Kommunikationsdaten

    unsigned short RecvType1;// Zuordnung der Antwortdaten Zusatz 1.
    unsigned short reserve1;// Reserve 1
    union rtdata1 { // Überwachungsdaten 1.
        POSE pos1;// Typ XYZ [mm/rad].
        JOINT jnt1;// Typ Gelenk [mm/rad].
        PULSE pls1; // Typ Impuls [mm/rad].
        long lng1[8]; // Typ Ganze Zahl [% / non-unit].
    } dat1;
    unsigned short RecvType2;// Zuordnung der Antwortdaten Zusatz 2.
    unsigned short reserve2;// Reserve 2
    union rtdata2 {//Überwachungsdaten 2.
        POSE pos2;// Typ XYZ [mm/rad].
        JOINT jnt2; // Typ Gelenk [mm/rad].
        PULSE pls2; // Typ Impuls [mm/rad] oder Typ Ganze Zahl [% / non-unit].
```

```
        long lng2[8];// Typ Ganze Zahl [% / non-unit].
    } dat2;
    unsigned short RecvType3;// Zuordnung der Antwortdaten Zusatz 3.
    unsigned short reserve3; // Reserve 3
    union rtdata3 {// Überwachungsdaten 3.
        POSE pos3;// Typ XYZ [mm/rad].
        JOINT jnt3;// Typ Gelenk [mm/rad].
        PULSE pls3;// Typ Impuls [mm/rad] oder Typ Ganze Zahl [% / non-unit].
        long lng3[8];// Typ Ganze Zahl [% / non-unit].
    } dat3;
} MXTCMD;
```

Quelldatei sample.cpp

```
// sample.cpp

#include <windows.h>
#include <iostream.h>
#include <winsock.h>
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <math.h>
#include "strdef.h"
#define NO_FLAGS_SET 0
#define MAXBUFLEN 512

INT main(VOID)
{
    WSADATA Data;
    SOCKADDR_IN destSockAddr;
    SOCKET destSocket;
    unsigned long destAddr;
    int status;
    int numsnt;
    int numrcv;
    char sendText[MAXBUFLEN];
    char recvText[MAXBUFLEN];
    char dst_ip_address[MAXBUFLEN];
    unsigned short port;
    char msg[MAXBUFLEN];
    char buf[MAXBUFLEN];
    char type, type_mon[4];
    unsigned short IOSendType;// Ein-/Ausgangssignalzuordnung für Datenübertragung
    unsigned short IORecvType;// Ein-/Ausgangssignalzuordnung der Antwortdaten
    unsigned short IOBitTop=0;
    unsigned short IOBitMask=0xffff;
    unsigned short IOBitData=0;
    cout << " Input connection destination IP address (192.168.0.1) ->";
    cin.getline(dst_ip_address, MAXBUFLEN);
    if(dst_ip_address[0]==0) strcpy(dst_ip_address, "192.168.0.1");
    cout << " Input connection destination port No. (10000) -> ";
    cin.getline(msg, MAXBUFLEN);
    if(msg[0]!=0) port=atoi(msg);
    else port=10000;

    cout << " Use input/output signal?([Y] / [N])-> ";
    cin.getline(msg, MAXBUFLEN);
    if(msg[0]!=0 && (msg[0]=='Y' || msg[0]=='y')) {
        cout << "What is target? Input signal/output signal([I]nput / [O]utput)-> ";
        cin.getline(msg, MAXBUFLEN);
```

```
            switch(msg[0]) {
                case 'O'://Ziel für das Ausgangssignal setzen
                case 'o':
                    IOSendType = MXT_IO_OUT;
                    IORecvType = MXT_IO_OUT;
                    break;
                case 'I':// Ziel für das Einganssignal setzen
                case 'i':
                default:
                    IOSendType = MXT_IO_NULL;
                    IORecvType = MXT_IO_IN;
                    break;
            }

            cout << " Input head bit No. (0 to 32767)-> ";
            cin.getline(msg, MAXBUFLEN);
            if(msg[0]!=0) IOBitTop = atoi(msg);
            else IOBitTop = 0;

            if(IOSendType==MXT_IO_OUT) { // Nur für Ausgangssignal
                cout << "Input bit mask pattern for output as hexadecimal (0000 to FFFF)-> ";
                cin.getline(msg, MAXBUFLEN);
                if(msg[0]!=0) sscanf(msg,"%4x",&IOBitMask);
                else IOBitMask = 0;
                cout << "Input bit data for output as hexadecimal (0000 to FFFF)-> ";
                cin.getline(msg, MAXBUFLEN);
                if(msg[0]!=0) sscanf(msg,"%4x",&IOBitData);
                else IOBitData = 0;
            }

        }
    cout <<" --- Input the data type of command. --- \n";
    cout <<"[0: None / 1: XYZ / 2:JOINT / 3: PULSE]\n".;
    cout <<" -- please input the number -- [0] - [3]->";
        cin.getline(msg, MAXBUFLEN);
        type = atoi(msg);

        for(int k=0; k<4; k++) {
            sprintf (msg," --- input the data type of monitor ( %d-th ) --- \n", k); .
            cout << msg;
            cout << "[0: None]\n";
            cout << "[1: XYZ / 2:JOINT / 3: PULSE] Command value \n";
            cout << "[4: XYZ/ 5: JOINT/ 6: PULSE] Command value after the filter process \n";
            cout << "[7: XYZ/ 5:JOINT/ 6:PULSE] Feedback value. \n";
            cout << "[10: Electric current value / 11: Electric current feedback] ... Electric current value. \n";
            cout << "Input the numeral [0] to [11] -> ";
            cin.getline(msg, MAXBUFLEN);
            type_mon[k] = atoi(msg);
        }
        sprintf(msg, "IP=%s / PORT=%d / Send Type=%d / Monitor Type0/1/2/3=%d/%d/%d/%d"
    , dst_ip_address, port , type
    , type_mon[0], type_mon[1], type_mon[2], type_mon[3]);
        cout << msg << endl;

        cout << "[Enter]= End / [d]= Monitor data display";
        cout << "[z/x]= Increment/decrement first command data transmitted by the delta amount. ";

        cout << " Is it all right? [Enter] / [Ctrl+C] ";
        cin.getline(msg, MAXBUFLEN);

    // Initialisierung der Windows-Socket-DLL
        status=WSAStartup(MAKEWORD(1, 1), &Data);
        if (status != 0)
        cerr << "ERROR: WSAStartup unsuccessful" << endl;
```

```
//Einstellung IP-Adresse, Schnittstelle usw.
    memset(&destSockAddr, 0, sizeof(destSockAddr));
    destAddr=inet_addr(dst_ip_address);
    memcpy(&destSockAddr.sin_addr, &destAddr, sizeof(destAddr));
    destSockAddr.sin_port=htons(port);
    destSockAddr.sin_family=AF_INET;

// Erstellung Socket
    destSocket=socket(AF_INET, SOCK_DGRAM, 0);
    if (destSocket == INVALID_SOCKET) {
        cerr << "ERROR: socket unsuccessful" << endl;
        status=WSACleanup();
        if (status == SOCKET_ERROR)
            cerr << "ERROR: WSACleanup unsuccessful" << endl;
        return(1);
    }

    MXTCMD MXTsend;
    MXTCMD MXTrecv;
    JOINT jnt_now;
    POSE pos_now;
    PULSE pls_now;

    unsigned long counter = 0;
    int loop = 1;
    int disp = 0;
    int disp_data = 0;
    int ch;
    float delta=(float)0.0;
    long ratio=1;
    int retry;
    fd_set      SockSet;// Mit Select ausgewählte Socket-Gruppe
    timeval     sTimeOut;// Einstellung Zeitüberschreitung (Timeout)

    memset(&MXTsend, 0, sizeof(MXTsend));
    memset(&jnt_now, 0, sizeof(JOINT));
    memset(&pos_now, 0, sizeof(POSE));
    memset(&pls_now, 0, sizeof(PULSE));

    while(loop) {
        memset(&MXTsend, 0, sizeof(MXTsend));
        memset(&MXTrecv, 0, sizeof(MXTrecv));

// Erstellung Übertragungsdaten
        if(loop==1) {// Nur für das erste mal
            MXTsend.Command = MXT_CMD_NULL;
            MXTsend.SendType = MXT_TYP_NULL;
            MXTsend.RecvType = type;
            MXTsend.SendIOType = MXT_IO_NULL;
            MXTsend.RecvIOType = IOSendType;
            MXTsend.CCount = counter = 0;
        }
        else {// Für das zweite mal und folgende
            MXTsend.Command = MXT_CMD_MOVE;
            MXTsend.SendType = type;
            MXTsend.RecvType = type*_mon[0];
            MXTsend.RecvType1= type_mon[1];
            MXTsend.RecvType2= type_mon[2];
            MXTsend.RecvType3= type_mon[3];

            switch(type) {
                case MXT_TYP_JOINT:
                    memcpy(&MXTsend.dat.jnt, &jnt_now, sizeof(JOINT));
                    MXTsend.dat.jnt.j1 += (float)(delta*ratio*3.141592/180.0);
```

```
                    break;
                case MXT_TYP_POSE:
                    memcpy(&MXTsend.dat.pos, &pos_now, sizeof(POSE));
                    MXTsend.dat.pos.w.x += (delta*ratio);
                    break;
                case MXT_TYP_PULSE:
                    memcpy(&MXTsend.dat.pls, &pls_now, sizeof(PULSE));
                    MXTsend.dat.pls.p1 += (long)((delta*ratio)*10);
                    break;
                default:
                    break;
            }
            MXTsend.SendIOType = IOSendType;
            MXTsend.RecvIOType = IORecvType;
            MXTsend.BitTop = IOBitTop;
            MXTsend.BitMask =IOBitMask;
            MXTsend.IoData = IOBitData;
            MXTsend.CCount = counter;
        }
// Tastatureingabe
// [Enter]=Ende / [d]= Anzeige der Überwachungsdaten oder keine/ [0/1/2/3]= Ändern der
   Überwachungsdatenanzeige
// [z/x]=Inkrementieren/Dekrementieren der ersten Befehlsdaten durch Übertragung des
   Differenzbetrags
while(kbhit()!=0) {
            ch=getch();
            switch(ch) {
            case 0x0d:
                MXTsend.Command = MXT_CMD_END;
                loop = 0;
                break;
            case 'Z':
            case 'z':
                delta += (float)0.1;
                break;
            case 'X':
            case 'x':
                delta -= (float)0.1;
                break;
            case 'C':
            case 'c':
                delta = (float)0.0;
                break;
            case 'd':
                disp = ~disp;
                break;
            case '0': case '1': case '2': case '3':
                disp_data = ch - '0';
                break;
            }
        }
        memset(sendText, 0, MAXBUFLEN);
        memcpy(sendText, &MXTsend, sizeof(MXTsend));
        if(disp) {
            sprintf(buf, "Send (%ld):",counter);
            cout << buf << endl;
        }
numsnt=sendto(destSocket, sendText, sizeof(MXTCMD), NO_FLAGS_SET
, (LPSOCKADDR) &destSokkAddr, sizeof(destSockAddr));
        if (numsnt != sizeof(MXTCMD)) {
```

```
            cerr << "ERROR: sendto unsuccessful" << endl;
        status=closesocket(destSocket);
        if (status == SOCKET_ERROR)
            cerr << "ERROR: closesocket unsuccessful" << endl;
        status=WSACleanup();
        if (status == SOCKET_ERROR)
            cerr << "ERROR: WSACleanup unsuccessful" << endl;
            return(1);
    }

    memset(recvText, 0, MAXBUFLEN);

    retry = 1;// Anzahl Empfangsversuche
    while(retry) {
        FD_ZERO(&SockSet);// Initialisierung SockSet
        FD_SET(destSocket, &SockSet);// Socket-Registrierung
        sTimeOut.tv_sec = 1;// Einstellung Zeitüberschreitung bei der Übertragung (sek)
        sTimeOut.tv_usec = 0;// (µsek)
        status = select(0, &SockSet, (fd_set *)NULL, (fd_set *)NULL, &sTimeOut);
        if(status == SOCKET_ERROR) {
            return(1);
        }

// Bei Empfang während der Zeitüberschreitung
        if((status > 0) && (FD_ISSET(destSocket, &SockSet) != 0)) {
            numrcv=recvfrom(destSocket, recvText, MAXBUFLEN, NO_FLAGS_SET, NULL, NULL);
            if (numrcv == SOCKET_ERROR) {
                cerr << "ERROR: recvfrom unsuccessful" << endl;
                status=closesocket(destSocket);
                if (status == SOCKET_ERROR)
                    cerr << "ERROR: closesocket unsuccessful" << endl;
                status=WSACleanup();
                if (status == SOCKET_ERROR)
                    cerr << "ERROR: WSACleanup unsuccessful" << endl;
                return(1);
            }
            memcpy(&MXTrecv, recvText, sizeof(MXTrecv));
            char str[10];
            if(MXTrecv.SendIOType==MXT_IO_IN)
        sprintf(str,"IN%04x", MXTrecv.IoData);
            else if(MXTrecv.SendIOType==MXT_IO_OUT)
    sprintf(str,"OT%04x", MXTrecv.IoData);
            else sprintf(str,"------");

        int DispType;
        void *DispData;

        switch(disp_data) {
            case 0:
                DispType = MXTrecv.RecvType;
                DispData = &MXTrecv.dat;
                break;
            case 1:
                DispType = MXTrecv.RecvType1;
                DispData = &MXTrecv.dat1;
                break;
            case 2:
                DispType = MXTrecv.RecvType2;
                DispData = &MXTrecv.dat2;
                break;
            case 3:
                DispType = MXTrecv.RecvType3;
                DispData = &MXTrecv.dat3;
```

```
                break;
            default:
                break;
        }

        switch(DispType) {
            case MXT_TYP_JOINT:
            case MXT_TYP_FJOINT:
            case MXT_TYP_FB_JOINT:
                if(loop==1) {
                    memcpy(&jnt_now, DispData, sizeof(JOINT));
                    loop = 2;
                }
                if(disp) {
                    JOINT *j=(JOINT*)DispData;
                    sprintf(buf, "Receive (%ld): TCount=%d Type(JOINT)=%d\n
%7.2f,%7.2f,%7.2f,%7.2f,%7.2f,%7.2f,%7.2f,%7.2f (%s)"
                        ,MXTrecv.CCount,MXTrecv.TCount,DispType
                        ,j->j1, j->j2, j->j3 ,j->j4, j->j5, j->j6, j->j7, j->j8, str);
                    cout << buf << endl;
                    }
                    break;
                case MXT_TYP_POSE:
                case MXT_TYP_FPOSE:
                case MXT_TYP_FB_POSE:
                    if(loop==1) {
                        memcpy(&pos_now, &MXTrecv.dat.pos, sizeof(POSE));
                        loop = 2;
                    }
                    if(disp) {
                        POSE *p=(POSE*)DispData;
                        sprintf(buf, "Receive (%ld): TCount=%d Type(POSE)=%d\n
%7.2f,%7.2f,%7.2f,%7.2f,%7.2f,%7.2f, %04x,%04x (%s)"
                            ,MXTrecv.CCount,MXTrecv.TCount,DispType
                                ,p->w.x, p->w.y, p->w.z, p->w.a, p->w.b, p->w.c
, p->sflg1, p->sflg2, str);
                        cout << buf << endl;
                    }
                    break;
                case MXT_TYP_PULSE:
                case MXT_TYP_FPULSE:
                case MXT_TYP_FB_PULSE:
                case MXT_TYP_CMDCUR:
                case MXT_TYP_FBKCUR:
                    if(loop==1) {
                        memcpy(&pls_now, &MXTrecv.dat.pls, sizeof(PULSE));
                        loop = 2;
                    }
                    if(disp) {
                        PULSE *l=(PULSE*)DispData;
                        sprintf(buf, "Receive (%ld): TCount=%d Type(PULSE/OTHER)=%d\n
%ld,%ld,%ld,%ld,%ld,%ld,%ld,%ld (%s)"
                            ,MXTrecv.CCount,MXTrecv.TCount,DispType
                            ,l->p1, l->p2, l->p3, l->p4, l->p5, l->p6, l->p7, l->p8, str);
                        cout << buf << endl;
                    }
                    break;
                case MXT_TYP_NULL:
                    if(loop==1) {
                        loop = 2;
                    }
```

```
                    if(disp) {
                        sprintf(buf, "Receive (%ld): TCount=%d Type(NULL)=%d\n (%s)"
                                ,MXTrecv.CCount,MXTrecv.TCount, DispType, str);
                        cout << buf << endl;
                    }
                    break;
                default:
                    cout << "Bad data type.\n" << endl;
                    break;
            }
            counter++;// Nur bei erfolgreicher Kommunikation hochzählen
            retry=0;// Schleife für Datenempfang verlassen
        }
        else { // Zeitüberschreitung Datenempfang
            cout << "... Receive Timeout! <Push [Enter] to stop the program>" << endl;
                retry--;// Rückwärtszähler Anzahl Versuche
                if(retry==0) loop=0; // Programm beenden, wenn Anzahl der Versuche „0“ ist
            }
        } /* while(Wiederholversuche) */
    } /* while(Schleife) */

// Ende
    cout << "/// End /// ";
    sprintf(buf, "counter = %ld", counter);
    cout << buf << endl;

//Socket schließen
    status=closesocket(destSocket);
    if (status == SOCKET_ERROR)
        cerr << "ERROR: closesocket unsuccessful" << endl;
    status=WSACleanup();
    if (status == SOCKET_ERROR)
        cerr << "ERROR: WSACleanup unsuccessful" << endl;

    return 0;

}
```

11.3 Stellungsmerker

Der Stellungsmerker kennzeichnet die Roboterstellung.
Bei einem 6-achsigen Roboter wird die Position der Handspitze in den Positionsdaten, bestehend aus X, Y, Z, A, B und C gespeichert. Trotzdem kann der Roboter mit den selben Positionsdaten verschiedene Stellungen einnehmen. Der Stellungsmerker kennzeichnet die Stellung eindeutig und wird als FL1 in der Positionskonstanten gespeichert (X, Y, Z, A, B, C) (FL1, FL2).

Die verschiedenen Arten der Stellungsmerker werden nachfolgend erläutert.

11.3.1 Beim Knickarmroboter

rechts/links

Beim 5-achsigen Roboter zeigt dieser Merker die Position des Handflanschmittelpunktes [P] in Bezug zu einer senkrechten Linie, die durch den Drehmittelpunkt der J1-Achse verläuft.

Beim 6-achsigen Roboter zeigt dieser Merker die Position des Drehmittelpunktes für die Handneigungsachse [Q] in Bezug zu einer senkrechten Linie, die durch den Drehmittelpunkt der J1-Achse verläuft.

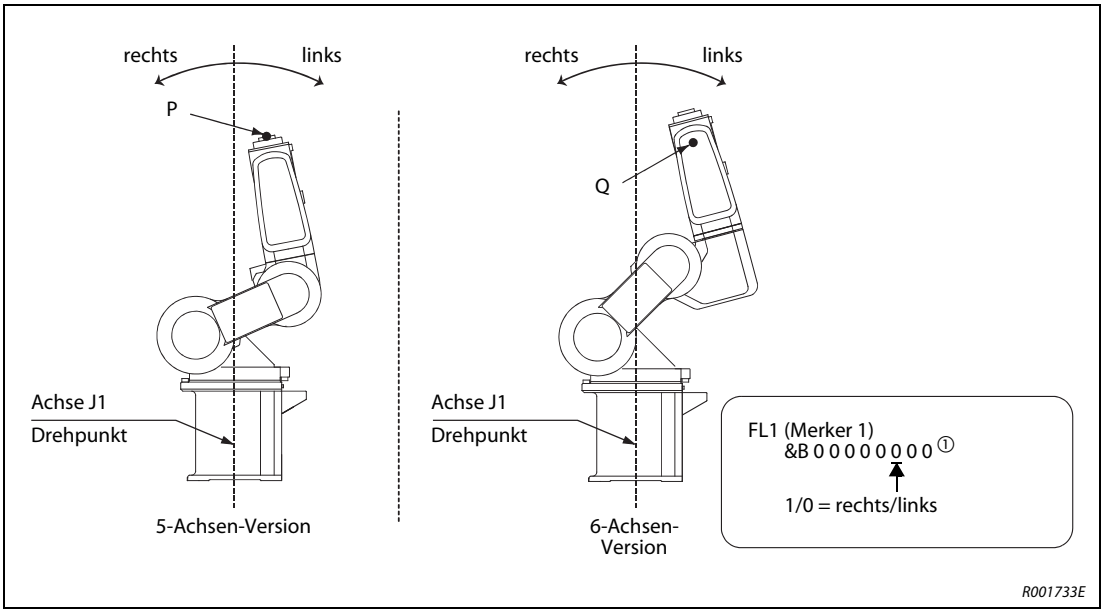


Abb. 11-7: *Stellungsmerker (rechts/links)*

① „&B“ kennzeichnet einen Binärwert

oben/unten

Dieser Merker zeigt die Position des Drehmittelpunktes für die Handgelenkneigung [Q] in Bezug zu einer Linie, die durch die Drehmittelpunkte der J2- und J3-Achse verläuft.

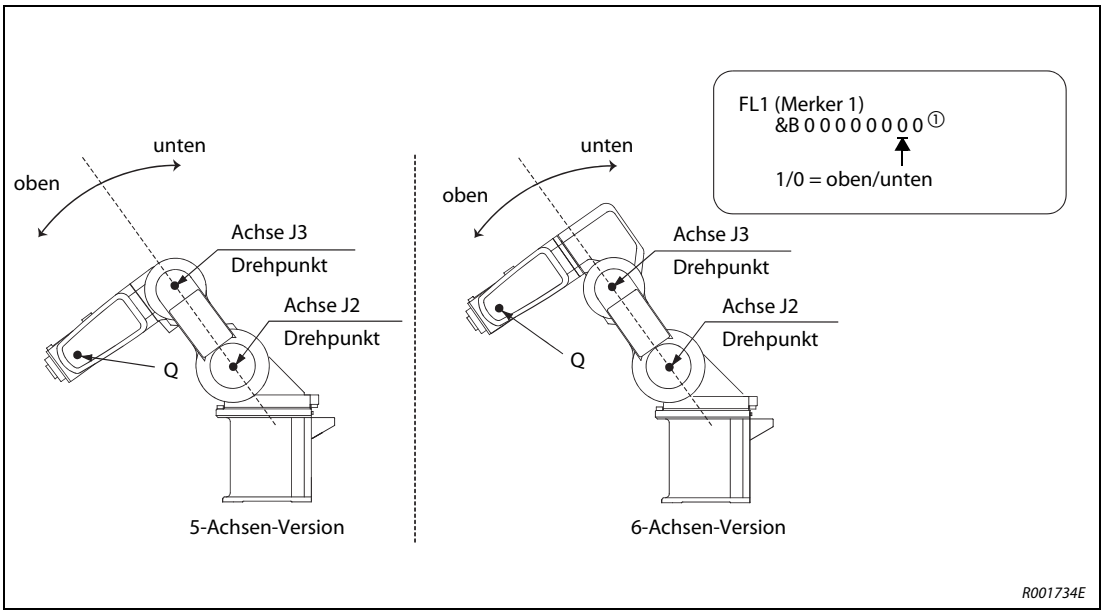


Abb. 11-8: *Konfigurationsmerker (oben/unten)*

① „&B“ kennzeichnet einen Binärwert

kippen/nicht kippen

Dieser Merker zeigt die Position des Handflanschmittelpunktes in Bezug zu der Linie, die durch die Drehmittelpunkte der J4- und J6-Achse verläuft.

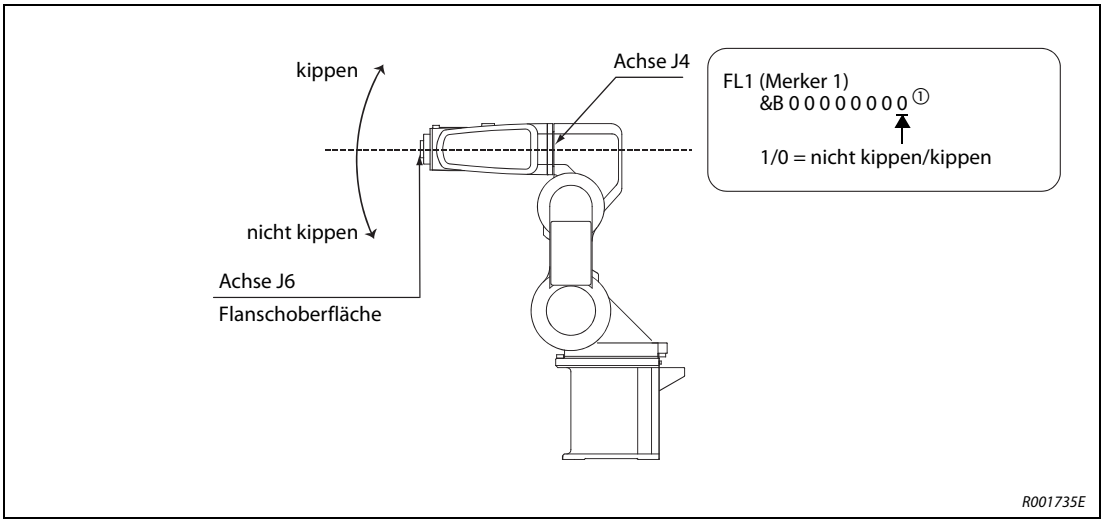


Abb. 11-9: *Konfigurationsmerker (kippen/nicht kippen)*

① „&B“ kennzeichnet einen Binärwert

11.3.2 Beim SCARA-Roboter

rechts/links

Es wird der Ort der Kugelumlaufspindel in Relation zu einer Linie bezeichnet, die durch die Drehmittelpunkte der J1- und J2-Achse verläuft.

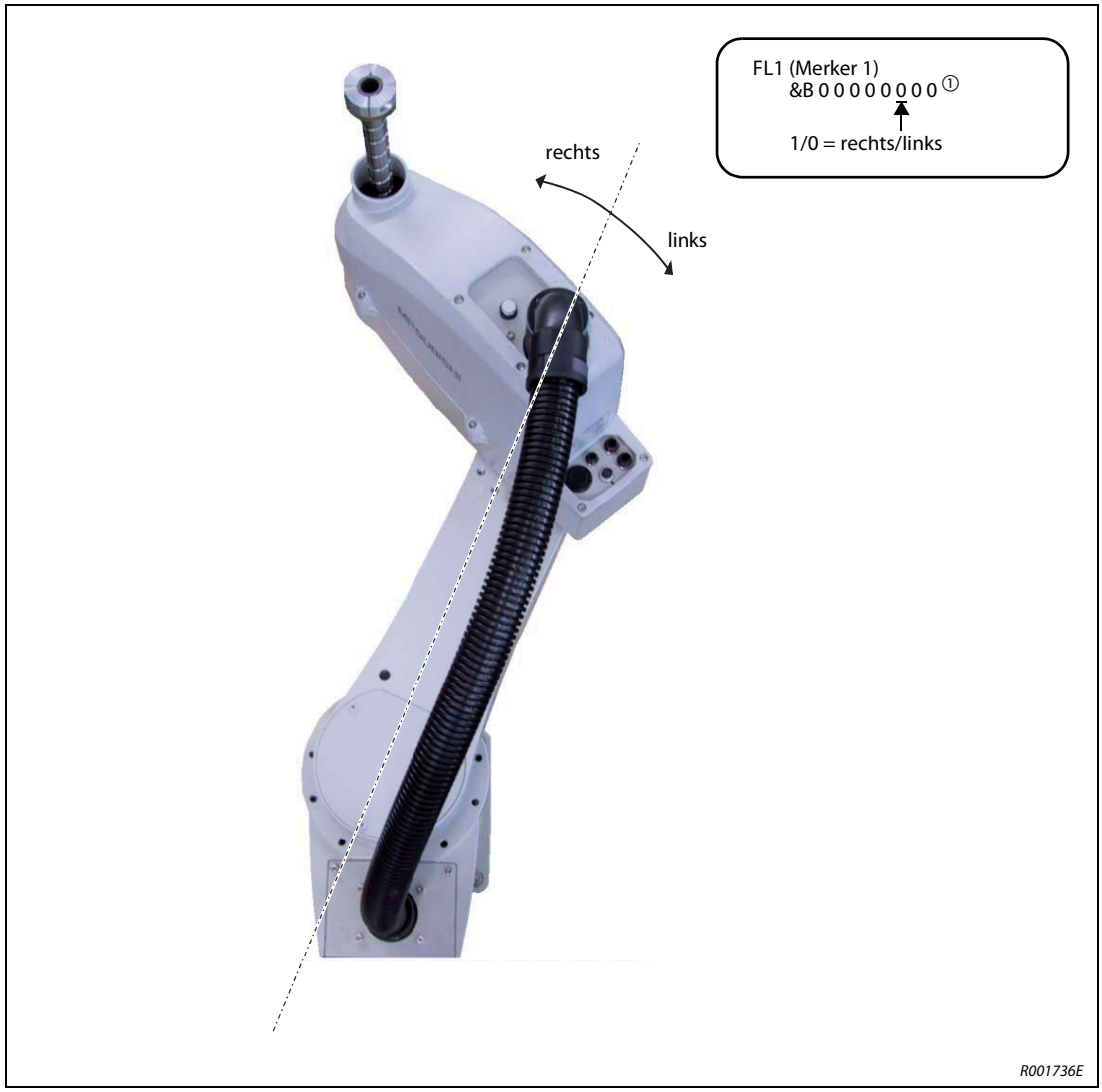


Abb. 11-10: *Konfigurationsmerker (rechts/links)*

① „&B“ kennzeichnet einen Binärwert

A Anhang

A.1 Fehlerdiagnose

Fehlernummer

Bei Auftreten eines Fehlers wird am Steuergerät eine 5-stellige Fehlernummer auf dem Display „STATUS.NUMBER“ angezeigt (z. B. C0010). Die LED auf dem RESET-Taster leuchtet.

Auf dem Display der Teaching Box erscheint eine 4-stellige Fehlernummer. Das erste Zeichen der Fehlernummer wird nicht angezeigt. Es erscheint z. B. „0010“ für „C0010“ und Klartext.

Im Menü „Fehlermeldungen anzeigen“ der Teaching Box kann eine Liste der bisher aufgetretenen Fehler aufgerufen werden. Dazu muss zuerst der Fehler zurückgesetzt werden.

In folgender Tabelle sind die Fehlernummern, die Fehlerursachen und die Gegenmaßnahmen aufgeführt. Lässt sich ein Fehler durch die aufgeführten Gegenmaßnahmen nicht beseitigen, setzen Sie sich mit Ihrem Vertriebspartner in Verbindung.

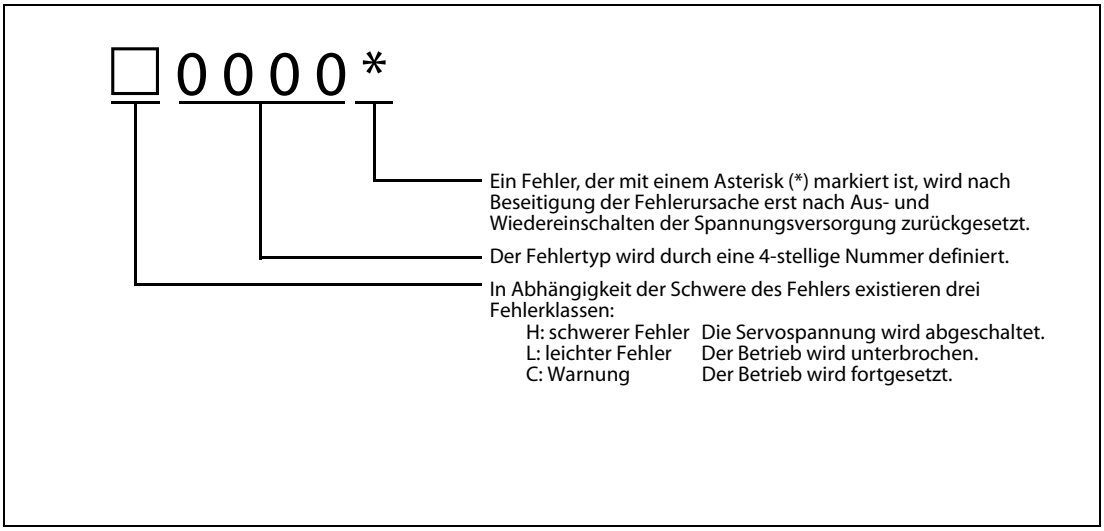

□ 0 0 0 0 *
Ein Fehler, der mit einem Asterisk (*) markiert ist, wird nach Beseitigung der Fehlerursache erst nach Aus- und Wiedereinschalten der Spannungsversorgung zurückgesetzt.
Der Fehlertyp wird durch eine 4-stellige Nummer definiert.
In Abhängigkeit der Schwere des Fehlers existieren drei Fehlerklassen:
H: schwerer Fehler Die Servospannung wird abgeschaltet.
L: leichter Fehler Der Betrieb wird unterbrochen.
C: Warnung Der Betrieb wird fortgesetzt.

Abb. A-1: *Aufbau einer Fehlermeldung*

HINWEIS

Die letzte Stelle der Fehlernummer kann eine Achsennummer anzeigen.
Bsp.: Die Fehlernummer H0931 bedeutet Überstrom des Motors der Achse Nr. 1.

Die Anzeige auf dem Bedienfeld des Steuergerätes erlischt

Erlischt die Anzeige auf dem Steuergerät (der Antriebseinheit), schalten Sie die Versorgungsspannung aus und wieder ein.
Erfolgt eine Fehlermeldung, ergreifen Sie die in der nachfolgenden Tabelle aufgeführten Gegenmaßnahme. Prüfen Sie die Liste der aufgetretenen Fehlermeldungen auch, wenn keine Fehlermeldung erfolgt und ergreifen Sie die entsprechende Gegenmaßnahme. Eine detaillierte Beschreibung, wie Sie die Liste der aufgetretenen Fehlermeldungen aufrufen, finden Sie in Abschn. 3.17.6.
Erlischt die Anzeige erneut, nachdem Sie die Gegenmaßnahmen ergriffen haben, kontaktieren Sie Ihren Vertriebspartner.

A.1.1 Übersicht der Fehlercodes

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H0001	Fehler beim Speichern einer Datei (SRVOFF)	Systemfehler	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0002	Fehler beim Speichern einer Datei (STOP)		
H0003	Systemfehler	Systemfehler	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0009*	Update (ALL)	Das System benötigt ein Update	Versorgungsspannung aus- und wieder einschalten
	Update (MAIN)	Das System benötigt ein Update	Versorgungsspannung ausschalten, Codierschalter auf „0" einstellen, Versorgungsspannung wieder einschalten
	Update (SERVO)	Das Servosystem benötigt ein Update	
	Servo-Software überschrieben	Servo-Software überschrieben	Versorgungsspannung aus- und wieder einschalten
C0010	Ungültige Dateiversion	Versionsunverträglichkeit	Datei wird automatisch initialisiert. Programm wird gelöscht.
C0011	Ungültige Version der Systemdaten	Versionsunverträglichkeit	Datei wird automatisch initialisiert. Versorgungsspannung aus und wieder einschalten
C0012	Initialisierung des Fehler-Logfiles	Versionsunverträglichkeit oder fehlerhaftes Fehler-Logfile	Fehler zurücksetzen und Betrieb fortsetzen
C0013*	Ungültige Datei	Die Daten bzw. Programme sind beschädigt.	MITSUBISHI-Vertriebspartner kontaktieren, um Initialisierung durchzuführen
C0014*	Systemfehler (ungültiger Mechanismus)	Die Zeichenkette darf nicht aus mehr als 16 Zeichen bestehen.	Namen korrigieren
C0015*	Ungültige Dateiversion	Ungültige Dateiversion	MITSUBISHI-Vertriebspartner kontaktieren
L0016*	Versorgungsspannung aus- und wieder einschalten	Die Zeit zwischen dem Aus- und Wiedereinschalten der Versorgungsspannung ist zu kurz	Die Zeit zwischen dem Aus- und Wiedereinschalten der Versorgungsspannung verlängern
C0018*	Ungültige Version (Speicher vergrößern)	Versionsabweichung	Zusatzspeicher wird initialisiert.
C0019*	Fehlerhafte Einstellung des Codierschalters auf der CPU	Codierschalter ist nicht auf „0" eingestellt.	Codierschalter auf „0" einstellen
H0020*	Der Backup-Datenname ist bereits vorhanden.	Fehlerhafte Backup-Daten	MITSUBISHI-Vertriebspartner kontaktieren
H0021*	Kein Anlegen weiterer Backup-Daten möglich	Überlauf des Steuerbereiches	
H0022*	Überlauf des Backup-Bereichs	Zu kleiner Bereich	
C0023*	Zusatzspeicher wurde hinzugefügt oder entfernt	Zusatzspeicher wurde hinzugefügt oder entfernt	Speicher prüfen
H0025*	Dateien werden in das ROM geschrieben.	Programme und Parameterdateien werden im ROM gespeichert.	Versorgungsspannung aus- und wieder einschalten
H0026*	Dateien werden aus dem ROM gelesen.	Programme und Parameterdateien aus dem ROM geladen.	
H0027*	Backup oder Wiederherstellung abgebrochen	Backup oder Wiederherstellung abgebrochen	
L0030	Handfehler Dreistufenschalter betätigen	Einstellfehler der Benutzerdaten	Nach Beseitigung der Fehlerursache Fehler zurücksetzen
L0031	Fehler in der Pneumatikversorgung der Hand	Einstellfehler der Benutzerdaten	

Tab. A-1: *Fehlercodes (1)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H0039	Die Signalleitung des Tür-Schließ-kontaktes ist defekt	Einer der beiden Kontaktpunkte des Tür-Schließkontaktes ist defekt oder die Verdrahtung ist nicht zweikreisig ausgeführt.	Schalten Sie die Spannungsversor-gung aus. Prüfen Sie die Verdrah-tung des Tür-Schließkontaktes. Stellen Sie sicher, dass die Verdrah-tung zweikreisig ausgeführt ist. Eine Detaillierte Beschreibung zum Anschluss des Tür-Schließkontak-tes finden Sie im Technischen Handbuch. Schalten Sie die Span-nungsversorgung wieder ein, nachdem Sie die Verdrahtung geprüft haben.
H0040	Das Signal vom Tür-Schließkon-takt wird geschaltet	Der Tür-Schließkontakt ist geöffnet	Prüfen Sie, ob das Eingangssignal des Tür-Schließkontaktes korrekt angeschlossen ist. Schließen Sie die Türe, die die Eingangssignale für den Türkontakt-Eingang schal-tet.
H0041*	CRC-Fehler auf dem externen E/A-Kanal 1	Fehler auf der Kommunikationslei-tung des externen E/A-Kanals 1	Prüfen Sie die Erdung und den Anschluss des Übertragungskabels
H0044	Falscher Anschluss des Schlüssel-schalters	Die Signalzustände auf den Leitun-gen stimmen nicht überein.	Schalten Sie die Spannungsversor-gung aus und wieder ein. Im Wie-derholungsfall MITSUBISHI-Ver-triebspartner kontaktieren
H0045	Fehler des Freigabeschalters der Teaching Box	Die Signalzustände auf den Leitun-gen stimmen nicht überein.	
H0046	Fehler des Zustimmschalters	Die Signalzustände auf den Leitun-gen stimmen nicht überein.	Schalten Sie die Spannungsversor-gung aus und überprüfen Sie die Verdrahtung. Der Anschluss muss zweikreisig ausgeführt werden (siehe auch Technisches Hand-buch).
H0050	Eingabe eines externen NOT-HALT-Signals	Das externe Stoppsignal wurde eingegeben. Wird der NOT-HALT-Schalter der Teaching Box betätigt kann dieser Fehler zusätzlich auf-treten.	Externen NOT-HALT zurücksetzen
H0051	Fehler im Anschluss des externen NOT-HALT-Kreises	Wird der NOT-HALT-Schalter der Teaching Box betätigt kann dieser Fehler zusätzlich auftreten.	Schalten Sie die Spannungsversor-gung aus und überprüfen Sie die Verdrahtung des NOT-HALT-Schal-ters. Stellen Sie sicher, dass die Ver-drahtung zweikreisig ausgeführt ist. Eine Detaillierte Beschreibung zum Anschluss des NOT-HALT-Schalters finden Sie im Techni-schen Handbuch. Schalten Sie die Spannungsversorgung wieder ein, nachdem Sie die Verdrahtung geprüft haben.
H0052	Eingabe des externen NOT-HALT-Signals für die Zusatzachse 1	Eingabe des externen NOT-HALT-Signals für die Zusatzachse 1	Prüfen Sie das NOT-HALT-Signal auf der Ansteuerkarte für die Zusatzachse.
H0053	Eingabe des externen NOT-HALT-Signals für die Zusatzachse 2	Das externe Stoppsignal wurde in den Zusatzverstärker eingegeben.	Prüfen Sie den NOT-HALT-Zustand des Zusatzverstärkers. Ist die Lei-tung des EM1-Signals (erzwungene Unterbrechung) für die Zusatzachse unterbrochen? NOT-HALT 1 und NOT-HALT 2 müssen separat ausgeführt sein. NOT-HALT 1 ist für die Schnittstellenkarte und NOT-HALT 2 für den Servoverstär-ker.
H0060	Eingabe des NOT-HALT-Signals über das Steuergerät	Das Stoppsignal wurde über das Steuergerät eingegeben.	NOT-HALT des Steuergeräts zurücksetzen

Tab. A-1: *Fehlercodes (2)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H0061	Fehler im Anschluss des NOT-HALT-Kreises vom Steuergerät	Die Signalleitung des NOT-HALT-Kreises ist instabil	Schalten Sie die Spannungsversorgung aus und überprüfen Sie die Verdrahtung des NOT-HALT-Schalters. Stellen Sie sicher, dass die Verdrahtung zweikreisig ausgeführt ist. Eine Detaillierte Beschreibung zum Anschluss des NOT-HALT-Schalters finden Sie im Technischen Handbuch. Schalten Sie die Spannungsversorgung wieder ein, nachdem Sie die Verdrahtung geprüft haben.
H0070	Eingabe des NOT-HALT-Signals über die Teaching Box	Eingabe des NOT-HALT-Signals über die Teaching Box oder es wurde der Schalter zum Lösen der Gelenkbremsen betätigt	NOT-HALT der Teaching Box zurücksetzen oder Schalter zum Lösen der Gelenkbremsen ausschalten Nach Zurücksetzen der Fehlermeldung Sicherung der Einheit TZ348 prüfen und falls defekt austauschen
H0071	Fehler im Anschluss des NOT-HALT-Kreises der Teaching Box	Die Signalleitung des NOT-HALT-Kreises ist instabil	Schalten Sie die Spannungsversorgung aus und überprüfen Sie die Verdrahtung des NOT-HALT-Schalters. Stellen Sie sicher, dass die Verdrahtung zweikreisig ausgeführt ist. Eine Detaillierte Beschreibung zum Anschluss des NOT-HALT-Schalters finden Sie im Technischen Handbuch. Schalten Sie die Spannungsversorgung wieder ein, nachdem Sie die Verdrahtung geprüft haben.
H0074	Fehler im System des [ENABLE/DIASABLE]-Schalters der Teaching Box	[ENABLE/DIASABLE]-Schalter arbeitet nicht korrekt.	Anschluss der Teaching Box prüfen. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0075	Übertragungsfehler der Teaching Box	Die Kommunikation zwischen Teaching Box und Steuergerät wurde unterbrochen.	
H0082*	Sicherung für die pneumatische Greifhand defekt	Sicherung für die pneumatische Greifhand defekt	Sicherung wechseln (siehe auch Technisches Handbuch)
H0083*	Sicherung der Spannungsversorgung der pneumatischen Greifhand defekt	Sicherung der Spannungsversorgung der pneumatischen Greifhand defekt oder Kurzschluss in der Spannungsversorgung für die Handeingangssignale	Anschluss des Handkabels prüfen oder Sicherung wechseln (siehe auch Technisches Handbuch)
H0084*	Sicherung des Steuergerätes defekt	Sicherung des Steuergerätes defekt	Sicherung wechseln
H0085*	Sicherung der Spannungsversorgung des externen NOT-HALT-Kreises defekt	Sicherung der Spannungsversorgung des externen NOT-HALT-Kreises defekt	
H0086	Überstrom der Schnittstellenkarte für die Greifhand	Handmotor oder Platine der motorbetriebenen Greifhand defekt	Handmotor oder Platine der motorbetriebenen Greifhand austauschen
H0087*	Sicherung der Bremsen defekt	Sicherung der Bremsen defekt	Sicherung wechseln
L0091	Signal ist bereits einem speziellen Ausgang zugeordnet.	Signal kann nicht 2-mal zugeordnet werden.	Ausgangsnummer oder zugewiesenen Parameter ändern
H0099	Die Servo-Version ist ungültig ** ** = korrekte Servo-Version	Alte Servo-Version installiert	Korrekte Servo-Version installieren MITSUBISHI-Vertriebspartner kontaktieren
H0100*	Temperatur des Steuergerätes zu hoch	Ventilator arbeitet nicht oder Filter ist verunreinigt Beim CR3n arbeitet der Ventilator in der R7CPU nicht.	Ventilator prüfen oder Filter reinigen bzw. austauschen Ventilator in der R7CPU im CR3n prüfen

Tab. A-1: *Fehlercodes (3)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L0101	Temperatur des Steuergerätes zu hoch	Ventilator arbeitet nicht oder Filter ist verunreinigt Beim CR3n arbeitet der Ventilator in der R7CPU nicht.	Ventilator prüfen oder Filter reinigen bzw. austauschen Ventilator in der R7CPU im CR3n prüfen
C0102	Temperatur des Steuergerätes zu hoch	Ventilator arbeitet nicht oder Filter ist verunreinigt Beim CR3n arbeitet der Ventilator in der R7CPU nicht.	Ventilator prüfen oder Filter reinigen bzw. austauschen Ventilator in der R7CPU im CR3n prüfen
H0120*	Kurzzeitiger Netzausfall	Ausfall der Netzspannung für 20 ms oder länger	Anschluss und Netzspannung überprüfen
H0130*	Initialisierungsfehler in der Verbindung mit den externen E/As	Fehlerhafte Verbindung zu den externen E/As	Prüfen Sie die Erdung und den Anschluss des Übertragungskabels
H0140 (nur CRnQ)	Sicherung defekt (Sicherheitseinheit)	Sicherung defekt (Sicherheitseinheit)	Sicherung wechseln
H0141	Fehlerhafte Verdrahtung des EMGIN-Anschlusses	Fehlerhafte Verdrahtung des EMGIN-Anschlusse	Überprüfen Sie die Verdrahtung des EMGIN-Anschlusses • Wird bei der Verdrahtung der NOT-HALT-Einrichtung (EMGIN-Anschluss) die interne Spannungsversorgung verwendet, dürfen die Klemmen 1-2 oder 5-6 des EMGIN-Anschlusses nicht angeschlossen werden. • Ist bei Verwendung einer externen Spannungsversorgung die externe Spannungsquelle angeschlossen? • Ist die Spannungsversorgung an Klemme 1 des EMGIN-Anschlusses kurzgeschlossen? • Die 24-V-Spannungsversorgung wird an die Klemmen 2 und 3 des EMG-Anschlusses gelegt. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
C0150	Nicht definierte Seriennummer des Roboters	Nicht definierte Seriennummer des Roboters	Korrekte Seriennummer eingeben Siehe auch Technisches Handbuch
H016m (m = 1~3)	Parallele E/A-Schnittstelle und parallele E/A-Einheit nicht gleichzeitig verwendbar	Die E/A-Kanalnummer der parallelen Schnittstelle und der parallelen Einheit überlappen sich	Entweder parallele E/A-Schnittstelle oder parallele E/A-Einheit wählen
H017m (m = 1~3)	Hardware-Fehler der parallelen E/A-Schnittstelle	Eventuell ist die Schnittstellenkarte defekt	Schnittstellenkarte wechseln
H018m (m = 1~3)	Sicherung der parallelen E/A-Schnittstelle defekt (Steckplatz m-Fn)	Sicherung der parallelen E/A-Schnittstelle defekt	Sicherung der parallelen E/A-Schnittstelle wechseln (siehe auch Technisches Handbuch)
C049n* (n: Achse, $1 \leq n \leq 8$)	Ventilatorfehler im Roboter	Ventilator im Roboter defekt	Ventilator im Roboter austauschen
H050n* (n: Achse, $1 \leq n \leq 8$)	Einstellfehler Codierschalter	Fehlerhafte Einstellung des Codierschalters zur Festlegung der beiden internen Zusatzachsenansteuerungen	Einstellung korrigieren
H0510*	Einstellung des Spannungswandlers fehlerhaft	Widersprüchliche Einstellung des Codierschalters des Spannungswandlers und des Parameters SVPTYP	
H0520*	Fehlerhafte Achseneinstellung	Die Einstellung der Achse eines Roboters ist auch einem weiteren Roboter zugewiesen.	

Tab. A-1: *Fehlercodes (4)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H053n* (n: Achse, 1 ≤ n ≤ 8)	Speicherfehler im Servoverstärker	Prüfsumme im Servoverstärker fehlerhaft	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H054n* (n: Achse, 1 ≤ n ≤ 8)	Systemfehler im Servoverstärker	Zeitüberschreitung bei der Verarbeitung der Daten im Servoverstärker	
H055n* (n: Achse, 1 ≤ n ≤ 8)	Systemfehler im Servoverstärker	Abweichung der magnetischen Pole des Encoders	
H056n* (n: Achse, 1 ≤ n ≤ 8)	Fehler im A/D-Wandler des Servoverstärkers	Fehler im A/D-Wandler des Servoverstärkers bei der Initialisierung	
H057n* (n: Achse, 1 ≤ n ≤ 8)	Encoderfehler (E^2PROM)	Encoderdaten im E^2PROM sind fehlerhaft	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H058n* (n: Achse, 1 ≤ n ≤ 8)	Encoderfehler (LED)	Die LED des Encoders ist defekt	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H059n* (n: Achse, 1 ≤ n ≤ 8)	Encoderfehler (Positionsdaten)	Fehlerhafte Positionsdaten innerhalb einer Umdrehung des Encoders	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H060n* (n: Achse, 1 ≤ n ≤ 8)	Kein Encodersignal 1	Es liegt kein Eingangssignal vom Encoder des Motors an.	
H061n* (n: Achse, 1 ≤ n ≤ 8)	Kein Encodersignal 2	Es liegt kein Eingangssignal vom Encoder der Maschine an.	
H062n* (n: Achse, 1 ≤ n ≤ 8)	Fehler auf der Servoplatine	Es ist eine Fehler auf der Platine des Servoverstärkers aufgetreten	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H063n* (n: Achse, 1 ≤ n ≤ 8)	Fehler des Servoverstärkers bei nicht verwendeter Achse	Fehler im Leistungsteil einer nicht zum Betrieb verwendeten Achse	
H064n* (n: Achse, 1 ≤ n ≤ 8)	Systemfehler (ABS CPU)	Fehler der CPU bei der linearen Skalierung der Absolutwertposition	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H065n* (n: Achse, 1 ≤ n ≤ 8)	Fehler der Absolutwertposition	Fehlermeldung bei der Überwachung der Absolutwertposition innerhalb des linearen Skalierungsbereichs	
H066n* (n: Achse, 1 ≤ n ≤ 8)	Fehler der Inkrementalposition	Fehlermeldung bei der Überwachung der relativen Position innerhalb des linearen Skalierungsbereichs	
H067n* (n: Achse, 1 ≤ n ≤ 8)	CPU-Fehler des seriellen Encoders	Fehler der CPU durch seriellen Encoder	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H068n* (n: Achse, 1 ≤ n ≤ 8)	LED-Fehler des seriellen Encoders	Impulse der LED des seriellen Encoders fehlerhaft	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0690*	Fehler im Bremskreis	Fehler eines Transistors oder Widerstandes im Bremskreis	Versorgungsspannung aus- und wieder einschalten

Tab. A-1: *Fehlercodes (5)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H0700*	Schaltfehler des Schützes	Schütz ist trotz READY OFF-Signal eingeschaltet.	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0710*	Schaltfehler des Überstromrelais der Spannungsversorgung	Überstromrelais der Spannungsversorgung schaltet sich nicht ein.	
H0711*	Schaltfehler des Überstromrelais der Spannungsversorgung	Entladerelais auf der Converterkarte schaltet sich nicht ein.	Versorgungsspannung aus- und wieder einschalten
H0720*	Fehlerhafte Spannungsversorgung (Watch Dog)	Zeitüberschreitung in der Spannungswandlerroutine	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0730*	Schaltfehler des Überstromrelais der Spannungsversorgung	Überstromrelais der Spannungsversorgung schaltet sich nicht aus.	
H0740*	Fehler in der Spannungsversorgung des Leistungskreises	Ladefehler der Kapazität Der externe NOT-HALT-Schalter ist fehlerhat.	Versorgungsspannung aus- und wieder einschalten Prüfen Sie, ob sich die Spannungsversorgung im zulässigen Bereich befindet und ob der NOT-HALT-Schalter korrekt angeschlossen ist. Im Wiederholungsfall prüfen, ob beim Auftreten des Fehlers die Servoversorgung oder die Spannungsversorgung eingeschaltet wird und MITSUBISHI-Vertriebspartner kontaktieren
H0742	Fehler in der Spannungsversorgung des Leistungskreises	Die Spannungsversorgung ist aufgrund des Sicherheitsrelais abgesunken.	Überprüfen Sie das Sicherheitsrelais. Überprüfen Sie die Verdrahtung des externen NOT-HALT-Schalters. Beispiele für den Anschluss des NOT-HALT-Schalters finden Sie im Technischen Handbuch. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0750*	Speicherfehler des Spannungswandlers	Fehler im Speicherkreis des Spannungswandlers	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0760*	Fehler der Spannungsversorgung	Fehler in Datenkommunikation mit der Spannungsversorgung	Versorgungsspannung aus- und wieder einschalten
H0770*	Keine Spannung	Spannungsversorgungskabel unterbrochen oder falsch angeschlossen	Kabel und Anschluss prüfen
H078n* (n: Achse, $1 \le n \le 8$)	Watch Dog des Servoverstärkers	Zeitüberschreitung bei der Datenverarbeitung des Servoverstärkers	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H079n* (n: Achse, $1 \le n \le 8$)	Leiterplattenfehler des Servoverstärkers	Fehler auf der Leiterplatte des Servoverstärkers	
H080n* (n: Achse, $1 \le n \le 8$)	Timerfehler im Servoverstärker	Fehler des Timers im Servoverstärker	
H081n* (n: Achse, $1 \le n \le 8$)	Unterspannung des Servoverstärkers	Die Spannung zwischen den Klemmen P und N ist auf einen Wert kleiner gleich 200 V abgefallen.	Prüfen der Spannungsversorgung
H0820* H082n* (n: Achse, $1 \le n \le 8$)	Erdschluss eines Servomotors	Erdschlussfehler vom Servomotor, defektes Motorkabel	Kabelanschluss prüfen Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H083n* (n: Achse, $1 \le n \le 8$)	Überspannung des Servoverstärkers	Die Spannung zwischen den Klemmen P und N ist auf einen Wert größer gleich 400 V angestiegen.	Prüfen der Spannungsversorgung Versorgungsspannung aus- und wieder einschalten
H0840*	Kurzzeitiger Spannungsausfall des Servoverstärkers	Es ist ein Spannungsausfall von mindestens 50 ms Dauer aufgetreten.	

Tab. A-1: *Fehlercodes (6)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H0850*	Offene Phase/fehlerhafter Spannungswert	Eingangsspannung entspricht nicht der eingestellten Spannung.	Eingangsspannung und Wahlschalter zur Einstellung der Eingangsspannung prüfen
H0860	Versorgungsspannung zu hoch	Die Versorgungsspannung zwischen L+ und L– ist größer als 410 V.	
H087n* (n: Achse, 1 ≤ n ≤ 8)	Thermischer Encoderfehler	Der integrierte Thermoschutz des seriellen Encoders wurde aktiviert	Versorgungsspannung aus- und nach einer Wartezeit wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H0880* H088n* (n: Achse, 1 ≤ n ≤ 8)	Überhitzung des Bremswiderstandes im Leistungsteil	Bremswiderstand wurde überhitzt.	
H089n* (n: Achse, 1 ≤ n ≤ 8)	Überhitzung des Servomotors	Thermosicherung des Motors oder des Encoders wurde aktiviert.	Versorgungsspannung aus- und nach einer Wartezeit wieder einschalten Erhöhen der Beschleunigungs-/Bremszeit (siehe auch Befehle Accel, Ovrd, Spd und Roboterstatusvariablen M_SetAdl, M_LdFact und Parameter JADL)
H090n* (n: Achse, 1 ≤ n ≤ 8)	Geschwindigkeitsüberschreitung beim Anfahren einer Absolutwertposition	Die Geschwindigkeit war beim Anfahren einer Absolutwertposition während der Initialisierung größer gleich 45 mm/s	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H091n* (n: Achse, 1 ≤ n ≤ 8)	Geschwindigkeitsüberschreitung des Servoverstärkers	Überschreitung der maximal zulässigen Motorgeschwindigkeit	
H0920* H092n* (n: Achse, 1 ≤ n ≤ 8)	Überstrom im Leistungsteil des Servoverstärkers	Überstrom im Servoverstärker, im Leistungsteil oder Kurzschluss im Motorkabel	
H093n* (n: Achse, 1 ≤ n ≤ 8)	Motorüberstrom	Der Strom durch den Motor ist zu groß, Fehler am Ausgang des A/D-Wandlers oder Motorkabel defekt	
H094n* (n: Achse, 1 ≤ n ≤ 8)	Überlast 1	Überschreitung der zulässigen Zeit für eine Überlast	Verringern der Beschleunigungs-/Bremszeit (siehe auch Befehle Accel, Ovrd, Spd und Roboterstatusvariablen M_SetAdl, M_LdFact und Parameter JADL)
H095n* (n: Achse, 1 ≤ n ≤ 8)	Überlast 2	Überschreitung des maximal zulässigen Ausgangsstroms für eine Sekunde oder länger	Last prüfen, Roboter darf nicht mit umliegenden Einrichtungen kollidieren
H096n* (n: Achse, 1 ≤ n ≤ 8)	Positionsabweichung 1	Bei eingeschaltetem Servo ist die eingestellte Abweichung der aktuellen Position zur Zielposition zu groß.	Last prüfen, Roboter darf nicht mit umliegenden Einrichtungen kollidieren Bei niedrigen Umgebungstemperaturen oder einem Betrieb nach einer langen Betriebspause, Anlaufphase mit einer niedrigen Geschwindigkeit oder Warmlaufbetrieb ausführen.
H097n* (n: Achse, 1 ≤ n ≤ 8)	Positionsabweichung 2	Bei ausgeschaltetem Servo ist die eingestellte Abweichung der aktuellen Position zur Ziel-Position zu groß.	Einwirkung externer Kräfte bei eingeschaltetem Servo prüfen
H098n* (n: Achse, 1 ≤ n ≤ 8)	Positionsabweichung 3	Fehlerhafter Anschluss des Motors	Motoranschluss prüfen Kein Motorstrom bei Auftreten der Fehlermeldung für Positionsabweichung 1
H099n* (n: Achse, 1 ≤ n ≤ 8)	Kollisionsüberwachung 0	Fehler durch Kollisionsüberwachung 0 bei G0	Kollisionsursache beseitigen
H100n* (n: Achse, 1 ≤ n ≤ 8)	Kollisionsüberwachung 1	Fehler durch Kollisionsüberwachung 1 bei G1	Kollisionsursache beseitigen

Tab. A-1: *Fehlercodes (7)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H101n* (n: Achse, 1 ≤ n ≤ 8)	Kollisionsüberwachung	Es ist eine Kollision aufgetreten.	• Bewegen Sie den Roboter nach einem Zusammenstoß im JOG-Betrieb von der Kollisionsstelle weg. In Abhängigkeit von der Schwere der Kollision kann dabei die Fehlermeldung erneut auftreten. Schalten Sie dann die Servoversorgung wieder ein und fahren Sie im JOG-Betrieb fort. Tritt der Fehler erneut auf, lösen Sie die Bremsen und bewegen Sie den Roboter manuell. • Tritt dieser Fehler ohne einen Zusammenstoß auf, stellen Sie die Ansprechschwelle der Kollisionsüberwachung ein. Spricht die Funktion im Automatikbetrieb fehlerhaft an, vergrößern Sie den Wert des Parameters COLLVL der entsprechenden Achse. Spricht die Funktion im JOG-Betrieb fehlerhaft an, vergrößern Sie den Wert des Parameters COLLVLJG der entsprechenden Achse. Vergrößern Sie den Wert nicht übermäßig, da damit auch die Ansprechemfindlichkeit sinkt. Das Risiko eines fehlerhaften Ansprechens der Funktion kann auch durch die korrekte Einstellung der Parameter HNDDATn und WRKDATn minimiert werden. • Bei zeitgleichem Auftreten einer Geschwindigkeitsüberschreitung wird die durch den Geschwindigkeitswechsel hervorgerufene Drehmomentschwankung eventuell als Kollision gewertet. Beseitigen Sie die anderen Fehlerursachen und wiederholen Sie die Bewegung. • Bei niedrigen Temperaturen oder nach einer langen Betriebspause kann ein fehlerhaftes Ansprechen der Funktion auch durch die hohe Viskosität der Schmiermittel auftreten. Starten Sie den Betrieb mit einer niedrigen Geschwindigkeit (Warmlaufphase) oder aktivieren Sie den Warmlaufbetrieb.
H102n* (n: Achse, 1 ≤ n ≤ 8)	Regenerative Überlast des Servoverstärkers der Zusatzachse	Überhitzung des regenerativen Bremswiderstandes	Prüfen der regenerativen Lastkapazität und der Parameter für die Zusatzachse
H1030*	Regenerative Überlast des Spannungswandlers	Die regenerative Kapazität des Spannungswandlers ist überschritten worden.	15 min bei eingeschalteter Versorgungsspannung warten, dann Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren Falls vorhanden, Bremswiderstand abklemmen
H104n* (n: Achse, 1 ≤ n ≤ 8)	Keine Kommunikation mit Encoder	Kein Kommunikationsaufbau zwischen Servoverstärker und Encoder; Encoderkabel oder Anschluss fehlerhaft	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H105n* (n: Achse, 1 ≤ n ≤ 8)	Keine Kommunikation mit Encoder	Kein Kommunikationsaufbau zwischen Servoverstärker und Encoder	Anschluss des Encoderkabels prüfen

Tab. A-1: *Fehlercodes (8)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H106n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler mit Encoder	Unterbrechung der Kommunikation zwischen Servoverstärker und Encoder	Anschluss des Encoderkabels prüfen
H107n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler mit Encoder	Unterbrechung der Kommunikation mit dem Encoder	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H108n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler des Servoverstärkers	Unterbrechung der Kommunikation zwischen Steuergerät und Servoverstärker aufgrund eines fehlerhaften Kabels oder Kabelanschlusses	
H1090* H109n* (n: Achse, 1 ≤ n ≤ 8)	Initialisierungsfehler des Servoverstärkers	Einstellungen der Achsen fehlerhaft (Parameter, Codierschalter)	
H1100*	Kommunikationsfehler des Servoverstärkers	Unterbrechung der Kommunikation zwischen Steuergerät und Servoverstärker aufgrund eines fehlerhaften Kabels oder Kabelanschlusses	
H111n* (n: Achse, 1 ≤ n ≤ 8)	RS232C-Kommunikationsfehler mit Servoverstärker	Kommunikationsfehler zwischen Servoverstärker und PC	Kabelanschluss und Kabel prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H112n* (n: Achse, 1 ≤ n ≤ 8)	Verlust der Absolutposition	Daten der Absolutposition sind gelöscht. Eventuell ist die Kapazität der Batterie im Roboterarm erschöpft.	Batterien tauschen und Nullpunkt nach der ABS-Methode einstellen (Da die Positionsdaten nach dem Einstellen des Nullpunktes mittels der ABS-Methode exakt wiederhergestellt werden können, ist ein erneutes Teachen nicht notwendig.)
H113n* (n: Achse, 1 ≤ n ≤ 8)	Encoderdaten fehlerhaft	Encoderdaten pro Umdrehung sind fehlerhaft.	Vorgang wiederholen und Umgebungsbedingungen prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H114n* (n: Achse, 1 ≤ n ≤ 8)	CRC-Fehler der Daten des Servoverstärkers	Prüfsummenfehler in den Daten zum Steuergerät	Kabelanschluss und Kabel prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H115n* (n: Achse, 1 ≤ n ≤ 8)	Fehlerhafter Befehlswert der Kommunikationsdaten des Servoverstärkers	Die Befehlsdaten für die Zielposition sind zu groß.	
H116n* (n: Achse, 1 ≤ n ≤ 8)	Fehler in der Datenlänge zum Servoverstärker	Die Länge der Daten vom Steuergerät ist fehlerhaft.	
H117n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsdaten zum Servoverstärker fehlerhaft	Die Daten vom Steuergerät sind fehlerhaft.	
H118n* (n: Achse, 1 ≤ n ≤ 8)	Fehler 1 der Rückmeldeimpulse des Servoverstärkers	Fehlende Impulse im Rückmeldesignal vom Encoder	Encoderkabel und Anschluss prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H119n* (n: Achse, 1 ≤ n ≤ 8)	Fehler 2 der Rückmeldeimpulse des Servoverstärkers	Abweichung zwischen den Rückmeldungen der Encoder von Motor und Maschine	

Tab. A-1: *Fehlercodes (9)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H1200*	CRC-Fehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein CRC-Fehler aufgetreten.	Kabelanschluss und Kabel prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H121n* (n: Achse, 1 ≤ n ≤ 8)	ID-Fehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein ID-Fehler aufgetreten.	
H122n* (n: Achse, 1 ≤ n ≤ 8)	Fehler der Achsennummer in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein Fehler der Achsennummer aufgetreten.	
H123n* (n: Achse, 1 ≤ n ≤ 8)	Sub-ID-Fehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein Sub-ID-Fehler aufgetreten.	
H1240*	Datennummernfehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein Datennummernfehler aufgetreten.	
H125n* (n: Achse, 1 ≤ n ≤ 8)	Parameterfehler des Servoverstärkers	Fehlerhafte Einstellung eines Parameters des Servoverstärkers	Parameter auf einen zulässigen Wert einstellen
H126n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler des Encoders	Kein Kommunikationsaufbau möglich	Kabelanschluss und Kabel prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H127n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler des Encoders	Fehlerhafte Übertragung der seriellen Daten einer Absolutwertposition	Kabelanschluss und Kabel prüfen Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H128n* (n: Achse, 1 ≤ n ≤ 8)	Formatfehler des Encodersignals	Fehlerhaftes serielles Datenformat der Absolutwertposition	
H129n* (n: Achse, 1 ≤ n ≤ 8)	Absolute Positionsabweichung des Servoverstärkers	Abweichung der Daten der Absolutwertposition beim Einschalten	Prüfen, ob die Achse aufgrund des Eigengewichts oder durch äußere Krafteinwirkung beim Einschalten bewegt wurde
H130n* (n: Achse, 1 ≤ n ≤ 8)	Abweichung der Positionsskalierung	Abweichung der Rückmeldeimpulse von Encoder und Maschinenskalierung	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H131n* (n: Achse, 1 ≤ n ≤ 8)	Abweichung des Positions-Offsets	Abweichung der Rückmeldeimpulse von Encoder und Maschinenskalierung	
H132n* (n: Achse, 1 ≤ n ≤ 8)	Fehler der Multirotationsdaten des Servoverstärkers	Fehlerhafte Multirotationsdaten des Encoders	Beim nächsten Einschalten der Versorgungsspannung werden die aktuellen Positionsdaten eventuell nicht richtig erkannt. Schalten Sie die Versorgungsspannung ein und stellen Sie die Grundposition mit der ABS-Methode ein. Auch bei Anzeige des Fehlers ist ein einwandfreier Betrieb bis zum nächsten Einschalten der Versorgungsspannung möglich. Erst dann ist eine Positionsabweichung möglich. Stellen Sie die Grundposition nach dem Einschalten mit der ABS-Methode ein, um die aktuellen Positionsdaten auf die Werte vor dem Auftreten der Fehlermeldung zu korrigieren.
H133n* (n: Achse, 1 ≤ n ≤ 8)	Niedrige Batteriespannung des Encoders	Batteriespannung für den Encoder ist abgesunken.	Batterie wechseln (siehe auch Technisches Handbuch)
H134n* (n: Achse, 1 ≤ n ≤ 8)	Regenerative Überlast des Servoverstärkers	Regenerative Energie des Zusatzachsenverstärkers von 80 % oder mehr	Prüfen der regenerativen Kapazität und des Parameters zur Einstellung der regenerativen Kapazität Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren Falls vorhanden, Bremswiderstand abklemmen

Tab. A-1: *Fehlercodes (10)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H135n* (n: Achse, 1 ≤ n ≤ 8)	Überlast des Servoverstärkers	Überlast von 95 % oder mehr	Last prüfen, Roboter darf nicht mit umliegenden Einrichtungen kollidieren
H136n* (n: Achse, 1 ≤ n ≤ 8)	Warnung des Zählers für Absolutwertposition	Fehlerhafter Encoderzähler	Batterie und Encoderkabel prüfen
H137n* (n: Achse, 1 ≤ n ≤ 8)	Parameterfehler des Servoverstärkers	Einstellung eines Parameters außerhalb des zulässigen Wertebereichs	Parameter auf einen zulässigen Wert einstellen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H138n* (n: Achse, 1 ≤ n ≤ 8)	Achse entfernen	Es wurde vom Steuergerät ein Befehl zum Entfernen der Achse ausgeführt.	Befehl abbrechen
H1390	NOT-HALT des Servoverstärkers	Der NOT-HALT-Schalter des Steuergerätes wurde betätigt	NOT-HALT-Zustand zurücksetzen
C1400	Kurzzeitige Überschreitung der regenerativen Energie des Servoverstärkers	Kurzzeitige Überschreitung der regenerativen Energie	Regenerative Kapazität prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
C1410	Kurzzeitiger Spannungsausfall des Servoverstärkers	Es ist ein Spannungsausfall von mindestens 25 ms Dauer aufgetreten.	Prüfen der Spannungsversorgung
C1420	Regenerative Überlast des Servoverstärkers	Regenerative Energie von 80 % oder mehr	Geschwindigkeit des Roboters verringern Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren Falls vorhanden, Bremswiderstand abklemmen
C1430	Hauptkreis des Servoverstärkers der Zusatzachse ausgeschaltet	Der Servo wurden bei ausgeschaltetem Hauptkreis eingeschaltet.	Hauptkreis einschalten
H144n* (n: Achse, 1 ≤ n ≤ 8)	Systemfehler 2 im Servoverstärker	Prozessorfehler	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H1450*	Kurzzeitige Unterbrechung der 24-V-DC-Spannungsversorgung	Unterspannung in der 24-V-DC-Spannungsversorgung	Anschluss CN22 prüfen
H1460*	Überstrom in der Spannungsversorgung	In der Spannungsversorgung des Leistungsteils ist ein Überstrom aufgetreten	Anschluss der Versorgungsspannung prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H1470*	Frequenzschwankung	Die Abweichung der Netzfrequenz ist zu groß	Frequenz der Versorgungsspannung prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H1480*	Fehler eines Parameters für die Versorgungsspannung	Fehlerhafte Einstellung eines Parameters für die Versorgungsspannung	Parameter korrigieren
H1490*	Überhitzung des Leistungsteils	Der Thermoschutz des Leistungsteils hat ausgelöst	Funktion des Ventilators auf der Rückseite des Leistungsteils prüfen Eine detaillierte Beschreibung zum Installationsort des Ventilators finden Sie im Technischen Handbuch.
H150n* (n: Achse, 1 ≤ n ≤ 8)	Falscher Servomotor	Die Kombination von Zusatzachsen-Servoverstärker und Servomotor ist nicht korrekt.	Prüfen Sie die technischen Daten von Servoverstärker und Servomotor
H151n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler in der USB-Verbindung zum Servoverstärker	Kommunikationsfehler zwischen Servoverstärker und PC	Anschluss der Kommunikationsleitung und Leitung prüfen
H152n* (n: Achse, 1 ≤ n ≤ 8)	Leistungsüberschreitung des Servomotors	Die ausgegebene Leistung des Servomotors überschreitet den zulässigen Wert	Drehzahl des Servomotors verringern
H154n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler zwischen den Einheiten	Übertragungsdaten zwischen den Servoverstärkern sind fehlerhaft	Anschluss der Kommunikationsleitung und Leitung prüfen

Tab. A-1: *Fehlercodes (11)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H1550*	Fehlerhaft Eingabe des NOT-HALTs	Fehlerhafte Eingabe des externen NOT-HALT-Signals	Versorgungsspannung aus- und wieder einschalten
H156n* (n: Achse, 1 ≤ n ≤ 8)	Positionsabweichung 4	Die Achse wurde während der Einschaltprozedur des Servoverstärkers bewegt	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H157n* (n: Achse, 1 ≤ n ≤ 8)	Unregistrierter Servofehler	Es ist ein unregistrierter Servofehler aufgetreten.	Kann der Fehler nicht zurückgesetzt werden, schalten Sie die Spannungsversorgung aus und wieder ein. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H158n* (n: Achse, 1 ≤ n ≤ 8)	Unregistrierte Servowarnung	Es ist eine unregistrierte Servowarnung aufgetreten.	Kann die Warnung nicht zurückgesetzt werden, schalten Sie die Spannungsversorgung aus und wieder ein. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H1600*	Kein Mechanismus definiert	Keiner der Mechanismen wurde definiert	Mindestens einen Mechanismus definieren Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H1610*	Bezeichnung des Mechanismus ungültig	Die Bezeichnung des Mechanismus ist ungültig oder nicht registriert.	Eingabe korrigieren Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
C1620	Roboternummer ungültig	Die im externen Kommunikationsprotokoll festgelegte Mechanismusnummer ist ungültig	Korrekte Mechanismusnummer einstellen
C1630	Während eines Servofehlers kann die Funktion Servo ON nicht ausgeführt werden.	Während eines Servofehlers kann die Servospannung nicht eingeschaltet werden.	Servofehler zurücksetzen und dann Servospannung einschalten
C1640	Bei ausgeschaltetem Dreistufenschalter kann die Funktion Servo ON nicht ausgeführt werden.	Bei ausgeschaltetem Dreistufenschalter kann die Servospannung nicht eingeschaltet werden.	Dreistufenschalter betätigen und dann Servospannung einschalten
C1650	Bei gelöster Bremse kann die Funktion Servo ON nicht ausgeführt werden.	Bei gelöster Bremse kann die Servospannung nicht eingeschaltet werden.	Bremsen aktivieren und dann Servospannung einschalten
C1660	Während des Servo-ON-Vorgangs kann die Funktion Servo ON nicht ausgeführt werden.	Während des Servo-ON-Vorgangs kann die Servospannung nicht eingeschaltet werden.	Servospannung nicht während des Servo-ON-Vorgangs einschalten
C1670	Während des Servo-OFF-Vorgangs kann die Funktion Servo ON nicht ausgeführt werden.	Der Servo-OFF-Vorgang ist aktiv.	Servospannung nicht während des Servo-OFF-Vorgangs einschalten
H1680	Zeitüberschreitung beim Servo-ON-Vorgang	Die Servos sind nicht innerhalb der zulässigen Zeit eingeschaltet worden.	Schalten Sie die Spannungsversorgung aus und wieder ein. Tritt der Fehler erneut auf, prüfen Sie, ob sich die Versorgungsspannung im zulässigen Bereich befindet und ob der externe NOT-HALT-Schalter richtig angeschlossen ist. Prüfen Sie auch die Versorgungsspannung der Zusatzachse. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H1681	Fehlerhaftes Abschalten der Servospannung	Die Servospannung wurde ungewollt abgeschaltet.	Servoverstärker prüfen Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren

Tab. A-1: *Fehlercodes (12)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H1682	Zeitüberschreitung Servospannung (Sicherheitsrelais)	Die Spannungsversorgung des Leistungskreises ist aufgrund des Sicherheitsrelais zu niedrig.	Überprüfen Sie das Sicherheitsrelais. Überprüfen Sie die Verdrahtung des externen NOT-HALT-Schalters. Beispiele für den Anschluss des NOT-HALT-Schalters finden Sie im Technischen Handbuch. Prüfen Sie, ob ein Fehler im Servoverstärker für eine Zusatzachse vorliegt. Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
C1690	Während der Dreistufenschalter ausgeschaltet ist, kann keine Bremse gelöst werden.	Es kann keine Bremse gelöst werden, während der Dreistufenschalter ausgeschaltet ist.	Dreistufenschalter einschalten und dann Bremse lösen
C1700	Bei aktiviertem NOT-HALT kann keine Bremse gelöst werden.	Es kann keine Bremse gelöst werden, wenn das NOT-HALT-Signal eingegeben wird.	NOT-HALT-Status aufheben und dann Bremse lösen
C1710	Im Servo-ON-Zustand kann keine Bremse gesteuert werden.	Bei eingeschalteter Servospannung kann keine Bremse gesteuert werden.	Servospannung ausschalten und dann Bremsen steuern
C1720	Während die Bremsen gelöst werden, kann kein weiterer Prozess zum Lösen der Bremsen ausgeführt werden.	Es kann kein weiterer Prozess zum Lösen der Bremsen ausgeführt werden, während die Bremsen gelöst werden.	Bremsen nicht lösen, während der Prozess „Bremsen lösen“ aktiv ist
C1730	Während die Bremsen aktiviert werden, kann kein weiterer Prozess zur Aktivierung der Bremsen ausgeführt werden.	Es kann kein weiterer Prozess zur Aktivierung der Bremsen ausgeführt werden, während die Bremsen aktiviert werden.	Bremsen nicht aktivieren, während der Prozess „Bremsen aktivieren“ aktiv ist
C1740	Servoparameter können nicht geändert werden.	Während der Einstellung von Parametern können keine weiteren Parameter eingestellt werden.	Einstellung wiederholen
C1750	Einstellung der Servoparameter ist fehlgeschlagen.	Es konnte keine Einstellung der Servoparameter vorgenommen werden.	
C1760	Ungültige Daten der Grundposition	Die Daten der Gundposition sind nicht korrekt eingestellt.	Korrekte Daten der Grundposition einstellen Prüfen Sie die Daten auf Fehler wie 1 (eins) und I (Alphabet) oder O (Alphabet) und 0 (null) usw.
C1770	Einstellung der Daten der Grundposition nicht abgeschlossen	Die Grundposition ist nicht eingestellt.	Einstellung wiederholen
C1780	Ungültige Achseneinstellung für Grundposition	Fehlerhafte Einstellung der Grundposition einer korrespondierenden Achse	Setzen der Grundposition der korrespondierenden Achsen, z. B. J3 und J4 beim RH-Roboter. Eine detaillierte Beschreibung zur Einstellung der Grundposition finden Sie im Technischen Handbuch.

Tab. A-1: *Fehlercodes (13)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
C1781	Keine Einstellung der Grundposition im Servo-ON-Zustand	Es wurde versucht, die Grundposition bei eingeschalteter Servospannung einzustellen.	Vor der Einstellung der Grundposition Servospannung ausschalten
H179n* (n: Achse, 1 ≤ n ≤ 8)	Fehlerhafte Einstellung der Verfahrweggrenzen	Einstellung des Parameters MEJAR ist fehlerhaft Die Einstellung liegt außerhalb des zulässigen Bereichs.	Der Wert des Parameters muss kleiner als +131072 oder größer als –131072 sein. Prüfen Sie den Wert bei Verwendung einer Zusatzachse.
H1800*	Fehlerhafte Einstellung der Verfahrweggrenzen für Absolutwertpositionierung	Einstellung des Parameters MEMAR ist fehlerhaft (Die –-Seite ist größer eingestellt als die +-Seite oder die Einstellung der Plus-Seite ist kleiner als „0“.)	Einstellung des Parameters MEMAR im zulässigen Bereich korrigieren
H1810*	Einstellfehler der benutzerdefinierten Grundposition	Einstellung des Parameters USERORG ist fehlerhaft	Einstellung des Parameters USERORG korrigieren
L182n (n: Achse, 1 ≤ n ≤ 8)	Positionsdaten stimmen nicht überein. Grundposition überprüfen	Positionsdaten bei ausgeschalteter Spannungsversorgung verändert	Überprüfung der Grundposition Bei Verschiebung, diese neu einstellen
L1830	Bereichsüberschreitung bei Ausführung des JRC-Befehls	Bereichsüberschreitung bei Ausführung des JRC-Befehls	Aktuelle Position und Einstellbereich prüfen
L184n (n: Achse, 1 ≤ n ≤ 8)	Fehlerhafte Einstellung des Parameters JRCQTT	Bereichsüberschreitung bei der Einstellung des Parameters JRCQTT	Bereichsüberschreitung bei der Einstellung des Parameters JRCQTT
C1850	Kurzzeitiger Netzausfall	Spannungsausfall von mehr als 20 ms Dauer	Spannungsversorgung prüfen
L1860	Ungültiger Parameter (TLC)	Die Einstellung des Parameters TLC für die Anfahrtsrichtung ist fehlerhaft	Einstellung des Parameters TLC korrigieren (= X/Y/Z)
L1861	Ungültiger Parameter (J1OFFSET)	Die Einstellung des Parameters J1OFFSET ist fehlerhaft	Korrigieren Sie das 2te Element des Parameters J1OFFSET
L1862	Ungültiger Parameter	Die Einstellung des Parameters für die Servo-Log-Daten ist fehlerhaft	Korrigieren Sie den Parameters für die Servo-Log-Daten

Tab. A-1: *Fehlercodes (14)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
C1871	Stillstand des Ventilators Nr. XX der J1-Achse	CR2DA/CR2QA: Ventilator 1: Der Ventilator zur Kühlung des Wandler-Verstärkers auf der Rückseite des Steuergerätes (der Antriebseinheit) rotiert nicht. Ventilator 2: Der Ventilator zur Kühlung der CPU auf der Rückseite des Steuergerätes (der Antriebseinheit) rotiert nicht. CR3D/CR3Q: Ventilator 1: Der Ventilator auf der Rückseite des Servoverstärkers 1 oben rotiert nicht. Ventilator 2: Der Ventilator auf der Rückseite des Servoverstärkers 1 unten rotiert nicht. Ventilator 6: Der Ventilator zur Belüftung des Steuergeräts (der Antriebseinheit) rotiert nicht. Ventilator 7: Der Ventilator auf der Rückseite des Spannungswandlers rotiert nicht.	Überprüfen Sie, ob der entsprechende Ventilator rotiert und tauschen Sie ihn aus, falls nicht. Eine detaillierte Beschreibung zum Austausch der Ventilatoren finden Sie im Technischen Handbuch.
C1872	Stillstand des Ventilators Nr. XX der J2-Achse		
C1873	Stillstand des Ventilators Nr. XX der J3-Achse	CR2DA/CR2QA: Ventilator 1: Der Ventilator zur Kühlung des Wandler-Verstärkers auf der Rückseite des Steuergerätes (der Antriebseinheit) rotiert nicht. Ventilator 2: Der Ventilator zur Kühlung der CPU auf der Rückseite des Steuergerätes (der Antriebseinheit) rotiert nicht. CR3D/CR3Q: Ventilator 1: Der Ventilator auf der Rückseite des Servoverstärkers 2 oben rotiert nicht. Ventilator 2: Der Ventilator auf der Rückseite des Servoverstärkers 2 unten rotiert nicht.	
C1874	Stillstand des Ventilators Nr. XX der J4-Achse	CR3D/CR3Q: Ventilator 1: Der Ventilator auf der Rückseite des Servoverstärkers 2 oben rotiert nicht. Ventilator 2: Der Ventilator auf der Rückseite des Servoverstärkers 2 unten rotiert nicht.	
C1875	Stillstand des Ventilators Nr. XX der J5-Achse	CR3D/CR3Q: Ventilator 1: Der Ventilator auf der Rückseite des Servoverstärkers 2 oben rotiert nicht. Ventilator 2: Der Ventilator auf der Rückseite des Servoverstärkers 2 unten rotiert nicht.	
C1876	Stillstand des Ventilators Nr. XX der J6-Achse	CR3D/CR3Q: Ventilator 1: Der Ventilator auf der Rückseite des Servoverstärkers 1 oben rotiert nicht. Ventilator 2: Der Ventilator auf der Rückseite des Servoverstärkers 1 unten rotiert nicht. Ventilator 6: Der Ventilator zur Belüftung des Steuergeräts (der Antriebseinheit) rotiert nicht. Ventilator 7: Der Ventilator auf der Rückseite des Spannungswandlers rotiert nicht.	

Tab. A-1: *Fehlercodes (15)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L188n* (n: Achse, 1 ≤ n ≤ 8)	Stillstand des Ventilators der Zusatzachse Jn	Der Ventilator im Verstärker der Zusatzachse ist evtl. defekt.	Ventilator austauschen Sehen auch im Handbuch für den Servoverstärker der Zusatzachse nach
L189n* (n: Achse, 1 ≤ n ≤ 8)	Die Drehzahl des Ventilators der Zusatzachse Jn sinkt	Der Ventilator im Verstärker der Zusatzachse ist evtl. defekt.	Ventilator austauschen Sehen auch im Handbuch für den Servoverstärker der Zusatzachse nach
C1910	Stillstand des Ventilators im Servoverstärker	CR1DA/CR1QA: Der Ventilator zur Kühlung des Servoverstärkers im Steuergerät (in der Antriebseinheit) rotiert nicht. CR2DA/CR2QA: Siehe Fehlermeldungen C1871 bis C1873 CR3D/CR3Q: Der Ventilator auf der Rückseite des Servoverstärkers, der in der Fehlermeldung angezeigt wird, rotiert nicht. Treten mehrere der Fehlermeldungen C1871 bis C1876 gleichzeitig auf, siehe Fehlermeldungen C1871 bis C1876.	Überprüfen Sie, ob der entsprechende Ventilator rotiert und tauschen Sie ihn aus, falls nicht. Eine detaillierte Beschreibung zum Austausch der Ventilatoren finden Sie im Technischen Handbuch.
C1920	Stillstand des Ventilators der Spannungsversorgung	CR2DA/CR2QA: Der Ventilator zur Kühlung des Wandler-Verstärkers auf der Rückseite des Steuergerätes (der Antriebseinheit) rotiert nicht. CR3Q: Der Ventilator auf der Rückseite des Spannungswandlers rotiert nicht.	
C1930	Stillstand des Ventilators der CPU	CR1DA/CR1QA/CR3D/CR3Q: Der Ventilator zur Kühlung der CPU im Steuergerät (in der Antriebseinheit) rotiert nicht. CR2DA/CR2QA: Der Ventilator zur Kühlung der CPU auf der Rückseite des Steuergerätes (der Antriebseinheit) rotiert nicht.	
C1940	Stillstand des Ventilators zur Belüftung	CR1DA/CR1QA/CR3D/CR3Q: Der Ventilator zur Belüftung des Steuergeräts (der Antriebseinheit) rotiert nicht.	
H195n	Fehler xx im Verstärker der Zusatzachse	Fehlermeldung des Verstärkers einer Zusatzachse	Eine detaillierte Beschreibung des Fehlers finden Sie im Handbuch des Servoverstärkers. „xx" kennzeichnet die Fehlernummer des Servoverstärkers (MR-J3-□B). (Die Methode zum Zurücksetzen des Fehlers hängt von der Fehlernummer ab.)
C196n	Warnung xx im Verstärker der Zusatzachse	Warnmeldung des Verstärkers einer Zusatzachse	Eine detaillierte Beschreibung der Warnung finden Sie im Handbuch des Servoverstärkers. „xx" kennzeichnet die Nummer der Warnmeldung des Servoverstärkers (MR-J3-□B).
L2000	Servospannung ist ausgeschaltet.	Der Roboter kann sich nicht bewegen, da die Servospannung ausgeschaltet ist	Servospannung einschalten und Neustart ausführen
L2010	Keine Impulsausgabe	Fehlerhafte Festlegung des Impulsausgangs	Programm korrigieren
L2020	Lesen externer Positionsdaten	Während des Lesens eines externen Befehls konnte ein Befehl nicht ausgeführt werden.	

Tab. A-1: *Fehlercodes (16)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L2030	Anforderung JOG-Betrieb nicht akzeptiert	Die Anforderung zum JOG-Betrieb wurde ausgeführt, aber nicht akzeptiert.	JOG-Betrieb deaktivieren
H2031*	Fehlerhafte Einstellung der Dimensionen	Einstellung der Parameter JOGTSJ oder JOGJSP ist fehlerhaft.	Dimension von 5 oder kleiner einstellen
H2040	Die geteachte Position ist nicht korrekt	Änderung der geteachten Position	Die Werkstückkoordinaten können für die geteachte Position nicht berechnet werden. Dafür kommen folgende Gründe in Frage: ● zwei Punkte befinden sich an derselben Position ● drei Punkte liegen auf einer Geraden Überprüfen Sie die oben genannten Punkte und korrigieren Sie die geteachte Position.
H209n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Überlagerungsbereichs n	Es wurde versucht, den Roboter aus dem benutzerdefinierten Bereich n zu bewegen.	Position korrigieren
H211n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung der Verfahrwegsbegrenzungsebene n	Es wurde versucht, den Roboter außerhalb der Verfahrwegsbegrenzungsebene n zu bewegen.	
H212n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung der Verfahrwegsbegrenzungsebene n	Es wurde versucht, den Roboter außerhalb der Verfahrwegsbegrenzungsebene n zu bewegen.	
H2129	Fehlerhafte Daten zur Einstellung der Verfahrwegsbegrenzungsebene	Daten zur Festlegung der Verfahrwegbegrenzungsebene sind fehlerhaft	Daten korrigieren
H213n (n: Achse, 1 ≤ n ≤ 8)	Geschwindigkeitsüberschreitung Jn-Achse	Geschwindigkeitsgrenzwert der Achse n wurde überschritten. In der Linear- oder Kreis-Interpolation kann in Abhängigkeit der Roboterstellung eine Geschwindigkeitsüberschreitung auftreten.	Verringern Sie die Geschwindigkeit über den Ovrd- oder Spd-Befehl oder ändern Sie die Roboterstellung. Bei der RH-3SDHR-Serie kann der Fehler auftreten, wenn die Funktion SPDOPT gesperrt ist. Geben Sie in diesem Fall die Funktion frei und wiederholen Sie den Vorgang. Tritt der Fehler erneut auf, verringern Sie die Geschwindigkeit über den Ovrd- oder Spd-Befehl. (Die Zykluszeit kann sich bei freigegebener SPDOPT-Funktion verlängern.)
H214n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Absolutwertbereiches der Jn-Achse in +-Richtung	Der Bereich des Absolutwerts der Achse n in +-Richtung wurde überschritten.	Fehler temporär zurücksetzen (siehe Abschn. 3.13) und Roboter im JOG-Betrieb in den zulässigen Bereich zurückfahren
H215n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Absolutwertbereiches der Jn-Achse in –-Richtung	Der Bereich des Absolutwerts der Achse n in –-Richtung wurde überschritten.	

Tab. A-1: *Fehlercodes (17)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H216n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Gelenkwinkels der Jn-Achse in +-Richtung	Der Verfahrwegbereich der Achse n für Gelenkbewegungen wurde in +-Richtung überschritten.	Position korrigieren
H217n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Gelenkwinkels der Jn-Achse in –-Richtung	Der Verfahrwegbereich der Achse n für Gelenkbewegungen wurde in –-Richtung überschritten.	
H2181	Überschreitung des orthogonalen Bereiches der Achse X in +-Richtung	Die orthogonale Bereichsgrenze der X-Achse wurden in +-Richtung überschritten.	
H2182	Überschreitung des orthogonalen Bereiches der Achse Y in +-Richtung	Die orthogonale Bereichsgrenze der Y-Achse wurden in +-Richtung überschritten.	
H2183	Überschreitung des orthogonalen Bereiches der Achse Z in +-Richtung	Die orthogonale Bereichsgrenze der Z-Achse wurden in +-Richtung überschritten.	
H2191	Überschreitung des orthogonalen Bereiches der Achse X in –-Richtung	Die orthogonale Bereichsgrenze der X-Achse wurden in –-Richtung überschritten.	
H2192	Überschreitung des orthogonalen Bereiches der Achse Y in –-Richtung	Die orthogonale Bereichsgrenze der Y-Achse wurden in –-Richtung überschritten.	
H2193	Überschreitung des orthogonalen Bereiches der Achse Z in –-Richtung	Die orthogonale Bereichsgrenze der Z-Achse wurden in –-Richtung überschritten.	
H2500	Fehlerhafte Encoderdaten	Die Encoderdaten sind fehlerhaft	Wiederholgenauigkeit und Umgebungsbedingungen prüfen
L2510	Umkehrung des Parameters für die Transportbandverfolgung EXCRGM*	Der Parameter für die Transportbandverfolgung EXCRGM* ist fehlerhaft	Prüfen Sie den Parameter EXCRGM* für die Transportbandverfolgung EXCRGM*
L2520	Bereichsüberschreitung eines Parameters für die Transportbandverfolgung	Parameter für die Transportbandverfolgung TRBUF liegt außerhalb des zulässigen Bereichs	Prüfen Sie den Parameter TRBUF für die Transportbandverfolgung
L2530	Kein Schreibbereich für Daten vorhanden	Kein Schreibbereich für Daten vorhanden	Lesen Sie die Daten über TRRD
L2540	Keine Daten zum Lesen vorhanden	Keine Daten zum Lesen vorhanden	Lesen Sie die Daten, indem Sie zuerst den Befehl TRWRT und dann den Befehl TRRD ausführen
L2560	Fehlerhafter Parameter für die Transportbandverfolgung	Fehlerhafter Parameter für die Transportbandverfolgung	Parameterwert prüfen
L2601	Startposition außerhalb des Verfahrwegbereiches	Startposition liegt außerhalb des Verfahrwegbereiches.	Position korrigieren
L2602	Zielposition außerhalb des Verfahrwegbereiches	Zielposition liegt außerhalb des Verfahrwegbereiches.	
L2603	Zwischenposition außerhalb des Verfahrwegbereiches	Zwischenposition liegt außerhalb des Verfahrwegbereiches.	
L2700	Cmp-Fehler (Modus)	Festgelegter Modus weicht vom aktuellen Modus ab.	Cmp-Off-Befehl ausführen und dann festlegen
C2710	Cmp-Fehler (Abweichung)	Abweichung bei Ausführung des Cmp-Befehls zu groß	Programm, Position o. Ä. so ändern, dass die Abweichung verringert wird.
C2720	Cmp-Fehler (Gelenkwinkel)	Der zulässigen Gelenkwinkel bei Ausführung des Cmp-Befehls wurde überschritten.	Positionsdaten korrigieren oder Abweichung verkleinern
H272n (n: Achse, 1 ≤ n ≤ 8)	Cmp-Fehler (Jn-Gelenkwinkel)	Der zulässigen Gelenkwinkel der Achse n wurde bei Ausführung des Cmp-Befehls überschritten.	
H273n (n: Achse, 1 ≤ n ≤ 8)	Cmp-Fehler (Geschwindigkeit der Jn-Achse)	Die zulässige Geschwindigkeit der Achse n wurde bei Ausführung des Cmp-Befehls überschritten.	Positionsdaten korrigieren oder Geschwindigkeit verringern
C2740	Cmp-Fehler (Konvertierung von Koordinaten)	Fehlerhafte Konvertierung von Koordinaten bei Ausführung des Cmp-Befehls	Positionsdaten korrigieren

Tab. A-1: *Fehlercodes (18)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
C2750	Keine Ausführung bei aktivierter Transportbandverfolgung möglich	Keine Ausführung bei aktivierter Transportbandverfolgung möglich	Zuerst Befehl Trk Off ausführen, dann Ausführung wiederholen
L2800	Positionsdaten fehlerhaft	Kann auftreten, wenn die Position außerhalb des Bewegungsbereiches des Roboters liegt	Position korrigieren
L2801	Positionsdaten der Startposition fehlerhaft	Kann auftreten, wenn die Startposition außerhalb des Bewegungsbereiches des Roboters liegt	
L2802	Positionsdaten der Zielposition fehlerhaft	Kann auftreten, wenn die Zielposition außerhalb des Bewegungsbereiches des Roboters liegt	
L2803	Positionsdaten der Hilfsposition fehlerhaft	Kann auftreten, wenn die Hilfsposition außerhalb des Bewegungsbereiches des Roboters liegt	
L2804	Die Funktion SPDOPT kann nicht ausgeführt werden.	Fährt der Roboter RH-3SDHR/SQHR durch den Bereich des singulären Punkts, kann keine Linear-Interpolation ausgeführt werden.	Verlegen Sie den Endpunkt, um die Interpolationsmethode zu ändern oder vermeiden Sie ein Durchfahren des Bereichs des singulären Punkts.
L2810	Stellungsmerker fehlerhaft	Der Stellungsmerker der Start- und der Endposition muss passen.	Positionsdaten korrigieren
H2820	Werte der Beschleunigung/ Abbremsung fehlerhaft	Die Werte zur Festlegung der Beschleunigung/Abbremsung sind zu klein.	Werte vergrößern
H2830	Systemfehler (fehlerhafte Einstellung des Befehlsparameters „Type“)	Der Befehlsparameter „Type“ des Mov-Befehls wurde auf einen unzulässigen Wert (z. B. „–1“) gesetzt	Wert korrekt einstellen (z. B. „0“, „1“ usw.)
H2840	Systemfehler (Interpolationsparameter)	Parameterdaten sind beschädigt	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H2850	Systemfehler (Interpolationsnorm)	Ungültige Norm Fehler bei der internen Verarbeitung	
H2860	Systemfehler (Interpolationsmethode)	Es wurde eine ungültige Interpolationsmethode verwendet.	
H2870	Systemfehler (undefinierte Interpolationsdaten)	Die Positionsdaten für die Interpolation wurden nicht definiert. Fehler bei der internen Verarbeitung	
H2880*	Systemfehler (zu wenig Speicherplatz)	Kein Speicherplatz für die Daten des Interpolationsprozesses	
H2890	Systemfehler (undefinierte Fehlernummer)	Bei der internen Verarbeitung wurde eine undefinierte Fehlernummer generiert.	
L2900	Systemfehler M00 bis M04	Fehler bei der internen Verarbeitung	
L3100	Speicherüberlauf	Der Speicherplatz nimmt kontinuierlich ab, wenn z. B. ein Sprung über GoSub ohne Rücksprung über Return oder ein Sprung aus einer For-Next-Schleife über einen GoTo-Befehl erfolgt.	Bei einem Sprung über GoSub Rücksprung über Return und in einer For-Next-Schleife keinen Sprung über GoTo programmieren
L3110	Bereichsfehler eines Befehlsparameters	Befehlsparameter liegt außerhalb des zulässigen Einstellbereiches.	Bereich prüfen und Eingabe wiederholen
L3120	Anzahl der Befehlsparameter fehlerhaft	Anzahl der Befehlsparameter entspricht nicht dem zulässigen Wert.	Anzahl der Befehlsparameter prüfen und korrigieren
L3130	Datei bereits geöffnet	Es wurde versucht, eine bereits geöffnete Datei zu öffnen.	Dateinummer prüfen und evtl. richtige Datei öffnen
L3140	Öffnen einer Datei nicht möglich	Die Datei kann nicht geöffnet werden.	

Tab. A-1: *Fehlercodes (19)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L3150	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „INPUT“ gesetzt ist	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „INPUT“ gesetzt ist	Dateinummer prüfen, Zugriffsmethode prüfen und Vorgang wiederholen
L3170	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „OUTPUT“ gesetzt ist	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „OUTPUT“ gesetzt ist	
L3180	Systemfehler (Einlesen von Felddaten nicht möglich)	Systemfehler (Einlesen von Felddaten nicht möglich)	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L3200	Datei kann nicht gelesen werden.	Ein Lesen der Datei ist nicht möglich.	Dateiinhalt prüfen
L3210	Variable kann nicht geschrieben werden.	Ein Schreiben der Variablen ist nicht möglich.	Schreibschutzstatus der Variablen prüfen
L3220	Zu viele Programmverschachtelungen	Die Anzahl der zulässigen Programmverschachtelungen von If-Bedingungen oder For-Anweisungen ist überschritten.	Programm korrigieren und Ausführung wiederholen
L3230	Anzahl der For- und Next-Anweisungen ist nicht identisch.	Anzahl der For- und Next-Anweisungen ist nicht identisch.	
L3240	Zu viele Programmverschachtelungen (For, While)	Die Anzahl der Programmebenen einer For- oder While-Schleife übersteigt 16.	Programm korrigieren
L3250	Die Anzahl der While- und When-Anweisungen ist nicht identisch.	Die Anzahl der While- und When-Anweisungen ist nicht identisch.	Programm korrigieren und Ausführung wiederholen
L3251	Die Anzahl der Sprungziele übersteigt 32.	Die Anzahl der Verzweigungen übersteigt 32.	
L3252	Die Anzahl der If- und EndIf-Anweisungen ist nicht identisch.	Die Anzahl der If- und EndIf-Anweisungen ist nicht identisch.	
L3253	Zu viele Programmverschachtelungen (If, EndIf)	Die Anzahl der Programmebenen übersteigt 8.	
L3254	Die Anzahl der Select- und End-Select-Anweisungen ist nicht identisch.	Die Anzahl der Select- und End-Select-Anweisungen ist nicht identisch.	
L3255	Die Anzahl der If- und Else-Anweisungen ist nicht identisch.	Die Anzahl der If- und Else-Anweisungen ist nicht identisch.	
L3260	Parallele Ausführung der festgelegten Programme nicht möglich	Parallele Ausführung der festgelegten Programme nicht möglich	Einzelne Programme auswählen und ausführen
L3270	Befehlsgröße überschritten	Befehlsgröße überschritten	Maximal 256 Zeichen verwenden
L3280	Ausführung ohne GetM-Befehl nicht möglich	Die Ausführung der Anweisung ist ohne GetM-Befehl nicht möglich, oder es wurde eine ungültige Mechanismusnummer festgelegt.	Anweisung nach Ausführung des RelM- und GetM-Befehls in einem anderen Programmplatz ausführen
L3281	Keine Ausführung im Betrieb	Keine Ausführung im Betrieb	Keine Ausführung im Betrieb
L3282	Starten nicht möglich	Fehlerhafte Programmwahl oder Attributeinstellung	Programm in den festgelegten Programmplatz laden oder Attributeinstellung korrigieren
L3285	Keine Ausführung möglich (Run oder Wait)	Keine Ausführung im Betrieb oder Wartezustand möglich	Programm zurücksetzen (Unterbrechung aufheben)
L3286	Leeres Programm	Start eines leeren Programms	Programm schreiben oder korrektes Programm wählen
L3287	Keine Ausführung möglich (ERROR oder ALWAYS)	Programm kann nicht bei Startbedingungen ERROR und ALWAYS gestartet werden.	Programm korrigieren
L3288	Keine Ausführung während einer Programmeditierung möglich	Keine Ausführung des Programms während einer Programmeditierung möglich	Editierung zuerst beenden und anschließend Programm ausführen
L3289	Das ausgewählte Programm (SLT*) existiert nicht.	Das in der Programmplatzliste ausgewählte Programm existiert nicht.	Programmplatzparameter korrigieren
L3290	Keine Ausführung der Systemanwendung möglich	Keine Ausführung der Systemanwendung möglich	Prüfen, ob eine andere Anwendung (Benutzeranwendung) ausgeführt wird

Tab. A-1: *Fehlercodes (20)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L3300	Keine Ausführung einer Benutzeranwendung möglich	Keine Ausführung einer Benutzeranwendung möglich	Prüfen, ob die Systemanwendung ausgeführt wird
L3310	Der Befehl XRun kann nicht ausgeführt werden, wenn das Programm bereits abgearbeitet wird.	Der Befehl XRun kann nicht ausgeführt werden, wenn das Programm bereits abgearbeitet wird.	Festgelegten Programmplatz stoppen und Programm ausführen
L3320	Der Befehl XRun kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XRun kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Programm korrigieren und ausführen
L3330	Der Befehl XStp kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XStp kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	
L3340	Der Befehl XRst kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XRst kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Ein Rücksetzen kann nur im Wartestatus erfolgen
L3350	Der Befehl XRst kann nicht ausgeführt werden, wenn das Programm abgearbeitet wird.	Der Befehl XRst kann nicht ausgeführt werden, wenn das Programm abgearbeitet wird.	Verarbeitung unterbrechen und Befehl ausführen
L3360	Der Befehl XLoad kann nicht ausgeführt werden (keine PSA)	Der Befehl XLoad kann nicht ausgeführt werden, wenn die Programmwahl nicht freigegeben worden ist.	Befehl XRst ausführen und Vorgang wiederholen
L3361	Das Programm kann nicht geladen werden (SLT*)	Das im Programmplatzparameter (SLTn) ausgewählte Programm existiert nicht.	Programmplatzparameter korrigieren
L3370	Der Befehl XClr kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XClr kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XClr kann nur bei einem ausgewählen Programm ausgeführt werden
L3380	Der Befehl XClr kann nicht ausgeführt werden (keine PSA)	Der Befehl XClr kann nicht ausgeführt werden, wenn die Programmwahl nicht freigegeben worden ist.	Programm zurücksetzen und Befehl erneut ausführen
L3390	Keine kreisförmige Palettierung möglich	Keine kreisförmige Palettierung möglich	Andere Palettendefinition wählen
L3400	Systemfehler (zu wenig Speicherplatz)	Die Speicherplatzkapazität ist überschritten	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L3500	Das mit Input angegebene Format ist fehlerhaft.	Der Typ der im Input-Befehl angegebenen Variablen passt nicht zu den empfangenen Daten.	Format prüfen
L3501	Fehlerhafte Empfangsdaten (EBREAD)	Der Typ der festgelegten Variablen passt nicht zu den empfangenen Daten.	Daten prüfen
L3600	Sprungziel existiert nicht.	Das im Befehl Def Act, On Com oder On GoTo angegebene Sprungziel existiert nicht.	Sprungziel prüfen
L3700	Verwendung einer nicht definierten Variablen	Verwendung einer nicht definierten Variablen	Variable vor Verwendung definieren und Startwert zuweisen
L3710	Zu viele Programmverschachtelungen durch den CallP-Befehl.	Es wurden mehr als 7 Programmaufrufe mit dem CallP-Befehl ausgeführt.	Verschachtelungstiefe reduzieren
L3720	Die Anzahl der RX- und NX-Anweisungen ist nicht identisch.	Die Anzahl der RX- und NX-Anweisungen ist nicht identisch.	Die Anzahl der RX- und NX-Anweisungen anpassen.
L3750	Fehlerhafte Positionen (Def Plt)	Die Multirotationsinformationen (FL2 der J1-/J4-Achse) weichen voneinander ab.	Position so korrigieren, dass die Multirotationsinformationen übereinstimmen
L3760	Fehlerhafte Positionen (Def Plt)	Zu große Änderung der J1 oder J4-Achse (DefPlt)	Definieren Sie die Positionen so, dass die Änderung kleiner ist.
L3810	Typ des Arguments fehlerhaft	Typ des Arguments in einer arithmetischen Operation, einer Einzelargument- oder Vergleichsoperation o.Ä. ist fehlerhaft	Korrekten Typ des Arguments festlegen

Tab. A-1: *Fehlercodes (21)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L3820	Nicht definierter Code	Evtl. ist eine Programm- oder Systemstatusvariable zerstört.	Stellen Sie die Daten mit Hilfe des Backups wieder her. Ist kein Backup verfügbar, muss das Programm neu geschrieben werden.
L3830	Ausführung des GetM-Befehls nicht möglich	Ausführung des GetM-Befehls nicht möglich	Prüfen, ob der festgelegte Roboter mit einem anderen Programmplatz verwendet wird
L3840	Return-Befehl ohne GoSub-Befehl	Ausführung des Return-Befehls ohne GoSub-Befehl	Programm prüfen
L3850	Verwendung einer nicht definierten Palette	Verwendung einer nicht definierten Palette	Palette vor Verwendung über den Befehl Def Plt definieren
L3860	Ungültige Positionsdaten festgelegt	Fehlerhafte Positionsdaten	Positionsdaten prüfen
L3870	Ungültige Mechanismusnummer festgelegt	Der festgelegte Mechanismus kann nicht definiert werden.	Mechanismusnummer korrigieren
L3880	Ungültige Programmplatznummer	Die festgelegte Programmplatznummer ist ungültig.	Korrekte Programmplatznummer eingeben
L3890	Systemfehler	Fehlerhafter Betriebsbefehl, evtl. ist das Programm zerstört	Stellen Sie die Daten mit Hilfe des Backups wieder her. Ist kein Backup verfügbar, muss das Programm neu geschrieben werden.
L3900	Ausführung des JRC-Befehls nicht möglich	Der Befehl ist über den Parameter JRCEXE deaktiviert und kann deshalb nicht ausgeführt werden	Parameter JRCEXE ändern und anschließend Befehl JRC ausführen
L3910	Ausführung des Befehls JRC 0 nicht möglich	Der Befehl JRC 0 kann für die festgelegte Achse nicht ausgeführt werden	Einstellung korrigieren
L3930	Ausführung des Befehls nicht möglich	Kollisionsüberwachung ist aktiviert	Kollisionsüberwachung deaktivieren (ColChk Off)
L3940	Ausführung des ColChk-Befehls nicht möglich	Kollisionsüberwachung wird durch einen andere Funktion blockiert	Blockierende Funktion aufheben
L3950	Ausführung des NOERR-Modus nicht möglich	Kein Interrupt-Aufruf über die Statusvariable M_ColSts freigegeben	Interrupt-Aufruf über die Statusvariable M_ColSts definieren und freigeben
L3960	Interrupt-Aufruf kann nicht aufgehoben werden	NOERR-Modus wird ausgeführt	Nach Aufheben des NOERR-Modus Interrupt deaktivieren
L3970	Ausführung des ColChk-Befehls nicht möglich	Kollisionsüberwachung über Parameter COL gesperrt	Kollisionsüberwachung über Parameter COL freigegeben
L3980	Einstellung der Variablen M_LdM nicht möglich	Hohe Verfahrweggenauigkeit ist aktiviert (Prec On)	Hohe Verfahrweggenauigkeit aufheben (Prec Off)
L3981	Ausführung des Prec-Befehls nicht möglich	Fehlerhafte Einstellung der Variablen M_LdM	Lastmodus 1 einstellen (M_LdM = 0)
L3982	Ausführung nicht möglich (singulärer Punkt)	Aktivierung der Funktion zum Durchfahren singulärer Punkte bei diesem Roboter nicht möglich	TYPE-Einstellung prüfen
		Cmp-Befehl wird ausgeführt	Deaktivieren Sie die Achsenweichheit (Cmp Off)
		Die Steuerung einer Zusatzachse ist aktiv.	Deaktivieren Sie die Steuerung der Zusatzachse
		Die Transportbandverfolgung ist aktiviert.	Transportbandverfolgung deaktivieren (Trk Off)
		Vorausholen von Daten ist aktiviert	Vorausholen von Daten ist deaktivieren
		Der Roboter ist Teil eines Multi-Mechanismus-Systems	Funktion zum Durchfahren singulärer Punkte nicht verwenden
		Befehl ColChk On wird ausgeführt	Kollisionsüberwachung deaktivieren (ColChk Off)
H3983	Ausführung der Bewegung bei aktivierter Cnt-Einstellung nicht möglich	Ein Stellungsmerker oder Gelenkwinkel weicht ab	Verwenden Sie den Dly-Befehl zur Positionierung

Tab. A-1: *Fehlercodes (22)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L3984	Keine Durchfahrt (singulärer Punkt)	Roboter passiert einen Punkt in der Nähe des singulären Punktes	Geteachte Position korrigieren
L3985	Keine Durchfahrt (singulärer Punkt)	Roboter passiert einen Punkt in der Nähe des singulären Punktes	Geteachte Position korrigieren
L3990	ColChk kann nicht verwendet werden	Der Roboter ist Teil eines Multi-Mechanismus-Systems	Sperren Sie die Funktion mit dem Parameter COL
L3991	Maintenance-Forecast-Funktion kann nicht genutzt werden.	Der Roboter ist Teil eines Multi-Mechanismus-Systems	Deaktivieren Sie die Funktion zur Überwachung der Wartungsintervalle
L4000	Systemfehler (Zeitüberschreitung)	Programmexterner Fehler	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L4100	Zu große Anzahl der registrierten Programme	Zu große Anzahl der registrierten Programme	Nicht verwendete Programme löschen
L4110	Zu wenig Speicherplatz	Die Anzahl der Programme und die Datenmenge sind größer als der verfügbare Speicherplatz.	Nicht verwendete Programme oder Daten löschen oder optionale Speichererweiterung installieren
L4120	Programmname zu lang	Die Länge des Programmnamens darf höchsten 12 Zeichen und 3 Zusatzzeichen betragen.	Länge des Programmnamens auf einen zulässigen Wert kürzen
L4130	Ungültiger Programmname	Programmname enthält ein ungültiges Zeichen.	Es dürfen nur Buchstaben und Ziffern verwendet werden.
L4140	Das festgelegte Programm wurde nicht gefunden.	Das festgelegte Programm wurde nicht gefunden.	Gültiges Programm festlegen oder festgelegtes Programm erstellen
L4150	Fehler im Dateiinhalt	Evtl. sind die Daten durch einen Spannungsausfall während des Schreibvorgangs zerstört	Datei löschen
L4160	Kein gültiges Roboterprogramm	Das ausgewählte Programm ist kein gültiges Roboterprogramm.	Gültiges Programm auswählen
L4170	Programm wird editiert.	Programm wird editiert.	Editiertes Programm schließen
L4180	Programm wird ausgeführt.	Programm wird ausgeführt.	Programm stoppen
L4190	Das Programm ist ausgewählt	Programmausführung wird vorbereitet.	Programm zurücksetzen
L4200	Es können keine Daten in die Datei geschrieben werden.	1: Schreibschutz aktiv oder 2: Speicherplatz zu niedrig	1. Schreibschutz aufheben 2. Nicht verwendete Dateien löschen
L4210	Die eingegebene Anweisung ist zu lang.	Die Länge der Anweisung darf höchstens 240 Zeichen betragen.	Länge des Anweisung auf einen zulässigen Wert kürzen
L4220	Syntaxfehler	Die eingegebene Anweisung weist einen Syntaxfehler auf.	Bei der Eingabe Syntax beachten
L4230	Zeilennummer fehlerhaft	Die angegebene Zeilennummer existiert nicht.	Inhalt prüfen und gültige Zeilennummer eingeben
L4240	Schreibgeschütztes Programm	Schreibgeschütztes Programm	Schreibschutz aufheben
L4250	Keine weiteren Zeilen oder Variablen	Die Anzahl der gelesenen Zeilen oder Variablen übersteigt den maximal zulässigen Wert.	Programme prüfen
L4300	Der eingegebene Variablenname ist zu lang.	Die maximale Länge eines Variablennamens beträgt 8 Zeichen.	Variablennamen auf 8 Zeichen kürzen
L4310	Ungültiges Zeichen	Es wurde ein anderes Zeichen als A bis Z oder 0 bis 9 verwendet.	Keine ungültigen Zeichen verwenden
L4320	Variable ist schreibgeschützt.	Variable ist schreibgeschützt.	1. Verwenden einer speicherbaren Variablen 2. Schreibschutz aufheben
L4330	Es wurde versucht, eine nicht lesbare Variable zu lesen.	Variable ist lesegeschützt.	1. Verwenden einer speicherbaren Variablen 2. Schreibschutz aufheben
L4340	Nicht deklarierte Variable	Die Variable ist nicht deklariert.	Variable deklarieren
L4350	Doppelte Deklaration einer Variablen	Bereits definierte Variablen können nicht mit der Dim- oder Def-Anweisung neu definiert werden.	1. Variablennamen ändern und definieren 2. Definierte Variable löschen

Tab. A-1: *Fehlercodes (23)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L4360	Dieselbe Variable kann nicht öfter als 65535-mal verwendet werden.	Bei P1 = P1 + P2 wird 2-mal auf P1 und 1-mal auf P2 zugegriffen.	Programm so umschreiben, dass die Zahl der Zugriffe auf dieselbe Variable vermindert wird.
L4370	Fehlerhaftes Feldelement	1. Feldelement liegt nicht im zulässigen Bereich. 2. Variable ist kein Feld.	1. Setzen Sie das Feldelement auf einen Wert zwischen 1 und der max. Anzahl der Feldelemente. 2. Verwenden Sie kein Feldelement.
L4380	In einer Anweisung verwendete Variablen können nicht gelöscht werden.	In einer Anweisung verwendete Variablen können nicht gelöscht werden.	Löschen Sie die Anweisung mit der verwendeten Variablen.
L4390	Fehler in der Kombination der Variablentypen	Die Typen der benutzerdefinierten externen Variablen sind unterschiedlich.	Variablentypen anpassen
L4400	Programmfehler	Unzulässiger Programminhalt	Programm löschen
L4410	Fehler in umnummerierten Daten	Fehler in umnummerierten Daten	Daten zurücksetzen
L4420	Es dürfen keine Zeilennummern größer als 32767 verwendet werden.	Die neue Zeilennummer oder Zeilennummernlücke überschreitet den zulässigen Wert.	Keine Zeilennummern über 32767 verwenden
L4430	Das gesuchte Zeichen konnte nicht gefunden werden.	Das gesuchte Zeichen konnte nicht gefunden werden.	Programm prüfen
L4440	Die Marke existiert bereits.	Eine bereits definierte Marke kann kein zweites Mal definiert werden.	1. Markennamen ändern oder 2. Definierten Markennamen löschen
L4460	Unzulässiger Befehlsparameter	Der Befehlsparameter liegt außerhalb des zulässigen Bereiches	Befehlsparameter im zulässigen Bereich festlegen
L4470	Zeilennummer unzulässig	Zeilennummer unzulässig	Marke verwenden
L4800	Systemfehler (Basisprogramm)	Das Basisprogramm des Systems kann nicht geöffnet werden oder ist in den Parametern nicht korrekt festgelegt.	Legen Sie das Basisprogramm des Systems korrekt in den Parametern fest.
L4810	Kein Zugriff auf eine benutzerdefinierte globale Variable	Der Parameter PRGUSR ist nicht korrekt eingestellt	Zur Verwendung benutzerdefinierter externer Variablen muss das Benutzerbasisprogramm im Parameter PRGUSR registriert sein.
L4811*	Mehrfachdeklaration einer globalen Variablen	Eine globale Systemvariable ist als benutzerdefinierte globale Variable deklariert worden	Programm korrigieren
L4820	Keine Programmeditierung möglich	Das Programm wurde während der Editierung geschlossen. Das kann der Fall sein, wenn z.B. beim Editieren des Programms über den PC der [ENABLE/DISABLE]-Schalter der Teaching Box betätigt wird.	Programmeditierung wiederholen
L4900	Systemfehler	Der Programmname zur internen Verarbeitung ist unzulässig	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L4910	Fehlerhafte Einstellung der Programmiermethode	Der Parameter RLNG ist falsch eingestellt (1: MELFA-BASIC IV, 2: MELFA-BASIC V)	Parameter RLNG zurücksetzen oder anderes Programm auswählen
L4920	Keine Backup-Daten im ROM	Es sind keine Backup-Daten im ROM-Speicher	Vor dem Betrieb Backup anlegen
H5000	Fehlerhaftes Einschalten des Teach-Modus	Der [ENABLE/DISABLE]-Schalter der Teaching Box wurde im Automatikbetrieb umgeschaltet.	Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf DISABLE oder wählen Sie den Teach-Modus.
L5010	Signal AUTOENA ist AUS	Signal der Freigabe des Automatikbetriebs AUS	Signal zur Freigabe des Automatikbetriebs einschalten oder Teach-Modus wählen.
L5100	Kein Programm ausgewählt	Für den festgelegten Programmplatz wurde kein Programm gewählt.	Programm für den festgelegten Programmplatz wählen.

Tab. A-1: *Fehlercodes (24)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L5110	Kein kontinuierlicher Betrieb möglich	Es wurde ein anderes Programm gewählt.	Wahl des richtigen Programms
L5120	Programm kann nicht gewählt werden.	Die Programmwahl des festgelegten Programmplatzes ist nicht freigegeben.	Programm zurücksetzen
L5130	Servospannung kann nicht eingeschaltet werden.	Der Vorgang Servo AUS ist aktiv.	Warten Sie, bis die Servospannung ausgeschaltet ist und schalten Sie dann die Servospannung ein.
L5140	Lesen einer Datei nicht möglich	Es wird gerade ein Lese- oder Editierungsvorgang ausgeführt.	Editierte Datei schließen und Daten nach Abschluss des Lesevorgangs lesen
L5150	Keine Ausführung, da Grundposition nicht eingestellt	Grundposition nicht eingestellt	Grundposition einstellen
L5200*	Parameterfehler (TASKMAX)	Anzahl der parallel ausgeführten Programme überschreitet den im Parameter TASKMAX festgelegten Maximalwert (Werkseinstellung: 8, Maximalwert: 32).	Anzahl der parallel ausgeführten Programme reduzieren oder Parameter TASKMAX ändern
L5210*	Parameterfehler (MECHAMAX)	Anzahl der definierten Mechanismen überschreitet den im Parameter MECHAMAX festgelegten Maximalwert.	Anzahl der definierten Mechanismen reduzieren
L5400	Roboter können nicht definiert werden.	Mechanismen können nicht definiert werden.	Unabhängige Roboternummer definieren
L5410	Ein nicht existierender Modus wurde festgelegt.	Es wurde ein anderer Modus als AUTOMATIC/MANUAL festgelegt.	MITSUBISHI-Vertriebspartner kontaktieren
L5420	Ungültige Programmplatznummer	Festlegung einer ungültigen Programmplatznummer	Gültigen Programmplatz festlegen
L5430	Ungültige Mechanismusnummer	Festlegung einer ungültigen Mechanismusnummer	Gültigen Mechanismus festlegen
L5600	Keine Ausführung im Fehlerfall möglich	Keine Ausführung im Fehlerfall möglich	Fehler zurücksetzen
C5610	Keine Ausführung bei eingeschaltetem STOP-Signal	Keine Ausführung bei Eingabe des Stopp-Signals möglich	Stopp-Signal ausschalten und Ausführung wiederholen
L5620	Keine Ausführung bei eingeschaltetem CSSTOP-Signal	Keine Ausführung bei Eingabe des Zyklus-Stopp-Signals möglich	Zyklus-Stopp-Signal ausschalten
L5630	Keine Ausführung bei eingeschaltetem SRVOFF-Signal	Keine Ausführung bei Eingabe des Signals Servo OFF möglich	Servo-OFF-Signal ausschalten
L5640	Keine Ausführung während des Betriebs	Keine Ausführung während des Betriebs	Betrieb stoppen und Ausführung wiederholen
L5650	Keine Ausführung während des Stopp-Vorgangs	Keine Ausführung während des Stopp-Vorgangs	Stopp-Vorgang beenden und Ausführung wiederholen
L5660	Editierung im Betrieb (inklusive Startbedingung ALWAYS)	Keine Editierung während des Betriebs (inklusive des kontinuierlichen Betriebs)	Programm stoppen und Ausführung wiederholen
C5670	Keine Ausführung im Wartezustand	Keine Ausführung im Wartezustand	Programm zurücksetzen
L5990	Systemfehler (ungültiger Befehl)	Es wurde ein nicht registrierter Befehl ausgeführt.	Korrekten Befehl ausführen
L6010	Ein ungültiger Befehl wurde gesendet.	Der Fehler kann auftreten, wenn Daten vor Öffnen des Kommunikationskanals gesendet wurden oder ein nicht registrierter Befehl ausgeführt wurde oder die Version der Software für das Steuergerät inkompatibel zu der Version der Programmier-Software ist.	Daten erst nach Öffnen des Kommunikationskanals senden oder Versionen abgleichen
L6020	Betriebsrecht nicht aktiviert	Es wurden keine Betriebsrechte zugewiesen.	Betriebsrecht zuweisen
L6030	Betriebsrecht zum Editieren nicht aktiviert	Es wurden keine Betriebsrechte zum Editieren zugewiesen.	Betriebsrecht zum Editieren zuweisen

Tab. A-1: *Fehlercodes (25)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L6040	Systemfehler (ungültige Gerätenummer)	Die Gerätenummer ist nicht registriert.	Gültige Gerätenummer angeben
C6050	Datei kann nicht geöffnet werden.	Die Blockdatei kann nicht geöffnet werden.	Datei prüfen und korrekte Datei festlegen
C6060	Teach-Modus nicht aktiviert	Parameter im Teach-Modus schreiben	Teach-Modus wählen und dann Parameter schreiben
C6070	Zeit kann nicht eingestellt werden.	Die Zeiteinstellung kann nur bei gestopptem Programm und ausgeschalteter Servoversorgung durchgeführt werden.	Betrieb unterbrechen, Servospannung ausschalten und anschließend Zeit einstellen
C6080	Text zu lang	Die Zeichenkette eines Kommunikationstextes überschreitet die maximal zulässige Länge	Überwachen Sie die Anzahl der Zeichen in einem Kommentar über verschiedene Einstellungen, z.B. über Parameter
C6090	Rücksetzen des Roboters durch Aus- und Einschalten nicht möglich	Rücksetzen des Roboters durch Aus- und Einschalten nicht möglich	Rücksetzen des Roboters durch Aus- und Einschalten nicht möglich
C6500	Die Kommunikationsleitung wurde nicht über das Anwendungsprogramm geöffnet.	Es wurde im Programm kein Open-Befehl ausgeführt.	Open-Befehl ausführen und dann „PRN“ senden
H6510	Fehlerhafter Schnittstellenparameter	Parameter für die RS232C-Schnittstelle fehlerhaft	Kommunikationsparameter korrigieren und Spannungsversorgung aus- und wieder einschalten
H6520	Fehlerhafter Schnittstellenparameter	Parameter für die RS422-Schnittstelle fehlerhaft	
H6530*	Fehlerhafte Einstellung des Parameters COMDEV	Fehlerhafte Einstellung des Parameters COMDEV	Einstellung des Parameters COMDEV korrigieren
L6600	Ungültige Signalnummer	Die angegebene Signalnummer wurde nicht definiert.	Korrekte Signalnummer angeben
L6610	Handsensorsignal kann nicht geschrieben werden.	Ungültiges Signal	Gültiges Signal verwenden
L6620	Eingangssignal kann nicht in den für den Roboter festgelegten Bereich geschrieben werden.	Ungültiges Signal	Gültiges Signal verwenden
L6630	Eingangssignal kann nicht geschrieben werden.	Aktueller Eingangssignalmodus	Pseudo-Eingangssignal setzen
H6640	Fehlerhafte Einstellung eines speziellen Parameters	Parametereinstellung fehlerhaft	Einstellung des Parameters korrigieren
H6641	Doppelte Definition eines speziellen Eingangssignals	Parametereinstellung fehlerhaft	
H6642	Das STOP-Signal ist fest dem Signal Nummer 0 zugewiesen	Parametereinstellung fehlerhaft	Speziellen Parameter STOP auf „0“ setzen
H6643	Fehlerhafte Einstellung eines speziellen Parameters	Parametereinstellung fehlerhaft	Die Startnummer muss kleiner als die Endnummer sein
L6650*	Doppelte Definition eines speziellen Ausgangssignals	Ungültige Parametereinstellung	Parameter korrigieren
L6651*	Doppelte Definition eines speziellen Ausgangssignals und des Parameters HANDTYPE	Ungültige Parametereinstellung	
L6660	Schreiben eines speziellen Ausgangssignals nicht möglich	Fehlerhafte Programmeinstellung	Programm korrigieren
L6670	Ungültiges Bitmuster zum Zurücksetzen der Ausgänge	Die Parameter wurden nicht in Gruppen von 8 gesetzt.	Parameter korrigieren
C6900	Start im Pseudo-Eingangssignal-Modus	Parametereinstellungen	Setzen Sie die Parameter bei Verwendung aktueller Signale zurück und schalten Sie die Versorgungsspannung aus und wieder ein.
C7000	Die zu kopierende Quelldatei kann nicht gefunden werden.	Die zu kopierende Quelldatei kann nicht gefunden werden.	Korrekten Dateinamen angeben
C7010	Die zu löschende Zieldatei kann nicht gefunden werden.	Die zu löschende Zieldatei kann nicht gefunden werden.	
C7020	Die Datei, die umbenannt werden soll, kann nicht gefunden werden.	Die Datei, die umbenannt werden soll, kann nicht gefunden werden.	

Tab. A-1: *Fehlercodes (26)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H7030*	Parameteränderungsdatei zu groß	Änderungskapazität zu groß	MITSUBISHI-Vertriebspartner kontaktieren
C7040	Änderung des Parameters unzulässig	Änderung des Parameters ist aus Sicherheitsgründen nicht zulässig.	
H7050	Dateifehler	Datei ist beschädigt.	
H7060*	System-RAM-Ressourcen erschöpft	Speicherkapazität wurde überschritten.	
C7070	Speicherkapazität erschöpft	Die Programme und Daten überschreiten die Speicherkapazität.	Nicht verwendete Programme oder Daten löschen
L7071	Zu wenig Speicher für die Ausführung der Funktion CTN „Programm fortsetzen“	Die Funktion benötigt mehr als 100 kB Speicherplatz	Nicht verwendete Programme löschen
C7080	Parameter kann nicht gelesen werden.	Parameter existiert nicht oder Passwort ist nicht korrekt.	1. Korrekten Parameternamen eingeben 2. Korrektes Passwort eingeben
C7081	Parameter kann nicht geschrieben werden.	Parameter existiert nicht oder Passwort ist nicht korrekt.	
C7090	Parameterkommentar fehlerhaft	Die zulässige Zeichenzahl des Parameterkommentars ist überschritten	Der Betrieb wird nicht beeinflusst und kann fortgesetzt werden.
L7099	Parameter „xxx“ kann nicht gelesen werden.	Parameter existiert nicht	Überprüfen Sie die Parameterbezeichnung
H7300*	Parameterdatei laden	Damit die Parameter wirksam werden, muss die Spannungsversorgung aus- und wieder eingeschaltet werden.	Spannungsversorgung aus- und wieder einschalten
C7310	Geänderte Variablen wurden nicht gespeichert	Spannungsausfall während der Programmausführung	Spannungsversorgung während der Programmausführung nicht ausschalten
L7311	Spannungsversorgung wurde beim Speichern einer Datei ausgeschaltet	Spannungsversorgung wurde beim Speichern einer Datei ausgeschaltet	Spannungsversorgung nicht während des Speichervorgangs ausschalten
L7332	Bei aktivierter CTN-Einstellung keine Umschaltung in den ROM-Modus möglich	Bei aktivierter CTN-Einstellung „Programm fortsetzen“ keine Umschaltung in den ROM-Modus möglich	RAM-Modus wählen und Vorgang wiederholen
L7340	Verwendung der CTN-Funktion im DRAM-Modus nicht möglich	Verwendung der CTN-Funktion im DRAM-Modus nicht möglich	
L7341	Bei aktivierter CTN-Einstellung keine Umschaltung in den DRAM-Modus möglich	Bei aktivierter CTN-Einstellung keine Umschaltung in den DRAM-Modus möglich	
L7342	Globale Erweiterung im DRAM-Modus nicht möglich	Globale Erweiterung im DRAM-Modus nicht möglich	
L7343	Im PRGGBL-Modus keine Umschaltung in den DRAM-Modus möglich	Im PRGGBL-Modus keine Umschaltung in den DRAM-Modus möglich	
C7410	Monatliche Inspektion fällig	Monatliches Inspektionsintervall abgelaufen	Führen Sie die monatliche Inspektion durch
C7420	3-monatliche Inspektion fällig	3-monatliches Inspektionsintervall abgelaufen	Führen Sie die 3-monatliche Inspektion durch
C7430	6-monatliche Inspektion fällig	6-monatliches Inspektionsintervall abgelaufen	Führen Sie die 6-monatliche Inspektion durch
C7440	Jahresinspektion fällig	12-monatliches Inspektionsintervall abgelaufen	Führen Sie die Jahresinspektion durch
C7500	Keine Batteriespannung	Batterie im Steuergerät leer	Batterie austauschen und Daten laden
C7510	Batteriespannung zu niedrig	Batterie im Steuergerät leer	Batterie austauschen (siehe Technisches Handbuch)
C7520	Gebrauchszeit der Batterie überschritten	Batterie leer	
C7530	Schmiermittel erneuern	Schmiermittel verbraucht	Schmiermittel erneuern
C7540	Zahnriemen erneuern	Lebensdauer des Zahnriemens abgelaufen	Zahnriemen austauschen

Tab. A-1: *Fehlercodes (27)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
H7810	Parameterfehler ETHERNET (NETIP, NETGW, NETPORT, NETPROC, NETLOGIN, NETPSSWD, NETTOUTR, NETTOUTS, MXTCOM1-3)	Parametereinstellungen fehlerhaft	Parametereinstellungen korrigieren
H7820	Zeitüberschreitung Befehl MXT/MXS	Die im Parameter MXTTOUT eingestellte Zeit wurde überschritten	Wert des Parameters MXTTOUT vergrößern
H7830	Keine ETHERNET-Schnittstellenkarte installiert oder Befehl nicht ausführbar	Keine ETHERNET-Schnittstellenkarte installiert oder Befehl nicht ausführbar	ETHERNET-Schnittstellenkarte installieren
H7840	Empfangene Daten bei Ausführung des Befehls Mxt ungültig	Befehlsparameter und Datentyp stimmen nicht überein	Befehl und gesendete Daten prüfen
L8300	Maximale Anzahl der GETPOS-Definitionen überschritten	GETPOS darf maximal 8-mal definiert werden	Gleiche Interrupt-Nummer verwenden oder ein nicht benötigtes Programm zurücksetzen
L8310	GETPOS nicht definiert	GETPOS nicht definiert	GETPOS definieren
L8320	Systemfehler (GETPOS)	Interne Daten der GETPOS-Funktion ungültig	Versorgungsspannung aus- und wieder einschalten
L8400	Ungültige CNT-Daten (Prec oder M_LdM)	Die CNT-Daten sind ungültig. Änderung auf Prec Off und M_LdM = 0	Prec-Modus und M_LdM aktivieren
L8600	Bilderfassungssensor nicht angeschlossen	Bilderfassungssensor nicht angeschlossen	Schnittstellennummer und Parameter prüfen
L8601	Keine Verbindung zur Bilderfassung	Parameter des Benutzernamens oder Passwort fehlerhaft	Parameter korrekt einstellen
L8602	Passwort fehlerhaft	Das Passwort des Benutzernamens ist fehlerhaft	Passwort korrekt einstellen
L8603	Parameter fehlerhaft	Parameter des Benutzernamens oder Passwort fehlerhaft	Parameter korrekt einstellen
L8610	Kommunikationsfehler	Kommunikation unterbrochen	Übertragungsleitung prüfen
L8620	Die Nummer des Bilderfassungssensors ist nicht korrekt.	Die Nummer wurde nicht über NVOPEN festgelegt.	NVOPEN-Befehl prüfen
L8621	Der Name des Bilderfassungsprogramms ist nicht korrekt.	Der Programmname besteht aus mehr als 15 Zeichen	Programname auf maximal 15 Zeichen beschränken
L8622	Kein Bilderfassungsprogramm vorhanden	Es wurde kein Bilderfassungsprogramm festgelegt	Name des Bilderfassungsprogramms prüfen
L8630	Zelle für die Erfassung der erkannten Objekte ungültig	Keine Erfassung in der festgelegten Zelle	Festlegung der Zelle prüfen
L8631	Festgelegte Zelle außerhalb des zulässigen Bereichs	Überschreitung des zulässigen Bereichs für die Zelle	Bereich der Zelle prüfen
L8632	Zeitüberschreitung der Bilderfassung	Keine Antwort von der Bilderfassung	Antwort-Wartezeit prüfen
L8633	Zeitüberschreitung der Bilderfassung (NVTRG)	Keine Antwort von der Bilderfassung (NVTRG)	Netzwerkauslastung verringern
L8634	Komma in Bereichsfestlegung der Zelle	Komma in Bereichsfestlegung der Zelle	Bereichsfestlegung der Zelle prüfen
L8635	Kein Komma in Bereichsfestlegung der Zelle	Kein Komma in Bereichsfestlegung der Zelle	Bereichsfestlegung der Zelle prüfen
L8636	Name der Marke für die Bilderfassung fehlerhaft	Im Bilderfassungsprogramm wurde kein Markenname für die Bilderfassung festgelegt	Korrekten Markennamen für die Bilderfassung festlegen
L8640	Der Bild-Trigger ist fehlerhaft	Der Bild-Trigger des Sensors ist falsch eingestellt.	Einstellung des Bild-Triggers korrigieren
L8650	Bilderfassung online schalten	Der Bilderfassungssensor ist offline	Bilderfassungssensor online schalten
L8660	Keine Zugriffsberechtigung	Der Benutzername autorisiert keinen Vollzugriff	Benutzernamen prüfen
L8670	Kein Neustart möglich	Start wurde ohne Reset ausgeführt	Programm zurücksetzen
L8700 (nur CRnQ)	Fehler beim Selbsttest des Multi-CPU-Systems	Ungültiger Parameter des Multi-CPU-Systems, CPU-Fehler usw.	Prüfen Sie die Details des Fehlers mit dem GX Developer.

Tab. A-1: *Fehlercodes (28)*

Fehler-code	Bedeutung	Ursache	Gegenmaßnahme
L8710	Parameterfehler eines E/A-Moduls im PC (m): Zielparameternummer 1 bis 4 (entsprechend QXYUNIT 1 bis 4)	Fehlerhafte Einstellung des E/A-Moduls im PC	Parameterwert prüfen
L8720	Sicherung defekt (E/A-Modul im PC (m)) (m): Zielparameternummer 1 bis 4 (entsprechend QXYUNIT 1 bis 4)	Sicherung defekt (E/A-Modul im PC (m))	Detaillierte Informationen finden Sie im Handbuch zum E/A-Modul des PCs
L8730	Entnahmefehler E/A-Modul (m) (m): Zielparameternummer 1 bis 4 (entsprechend QXYUNIT 1 bis 4)	E/A-Modul wurde aus dem PC entfernt.	Setzen Sie das E/A-Modul in den PC ein.
H8800	Kommunikationsfehler des ASICs	Fehler beim Datenempfang vom Servoverstärker	Anschluss der Übertragungsleitung prüfen
	CRC-Fehler des ASICs	CRC-Fehler in den empfangenen Daten vom Servoverstärker	
	Datenbereichsfehler des ASICs	Ungültige Empfangsdaten vom Servoverstärker	
	Datenüberlauffehler des ASICs	Ungültige Empfangsdaten vom Servoverstärker	
	Anschlussfehler des ASICs	Keine Verbindung über die Übertragungsleitung	
	0-Byte-Übertragung des ASICs	Leeres Datenpaket wurde an den Servoverstärker übertragen	Versorgungsspannung aus- und wieder einschalten
H8810	Kommunikationsfehler	Fehler beim Senden von Daten zum Servoverstärker	Anschluss der Übertragungsleitung prüfen
	Ungültige ID	Die ID der Daten zur Kommunikation mit dem Servoverstärker ist ungültig	Versorgungsspannung aus- und wieder einschalten
H8820	Antwort-Wartezeit überschritten	Bei der Kommunikation mit dem Servoverstärker wurde die Antwort-Wartezeit überschritten	
H8830	Keine Verbindung zum Servoverstärker	Der Servoverstärker wird nicht erkannt	Anschlusskabel und Versorgungsspannung des Servoverstärkers überprüfen
	Fehlerhafte Einstellung einer Servoachse	Servoachse nicht korrekt eingestellt	Einstellung der Servoachse überprüfen (Parameter, Codierschalter)
	Falscher Servoverstärker	Es ist ein falscher Servoverstärker angeschlossen.	Anschluss zum falschen Servoverstärker lösen
	Falsches Kommunikationsprotokoll	Falsches Kommunikationsprotokoll	Anschluss zum falschen Servoverstärker lösen
	Ungültige Motor-ID	Es ist ein falscher Servovmotor angeschlossen.	Anschluss zum falschen Servomotor lösen
	Fehler beim Einstellen eines Parameters	Der eingestellte Wert eines Parameters für den Servoverstärker ist fehlerhaft.	Korrekten Wert einstellen
	Nummer des Servoverstärkers ungültig	Die Nummer des angeschlossenen Servoverstärkers ist ungültig.	Nummer des angeschlossenen Servoverstärkers prüfen
H8870	Vorübergehender Verarbeitungsfehler	Falsche Anforderungsdaten übertragen	Anforderungsdaten prüfen
	Vorübergehende Zeitüberschreitung bei der Verarbeitung	Keine Antwort des Servoverstärkers auf die Anforderung	Senden Sie die Anforderung erneut.
H9000 : H9099	Ein schwerer benutzerdefinierter Fehler ist aufgetreten.	Das Roboterprogramm hat einen schweren Fehler gemeldet.	Programm prüfen
L9100 : L9199	Ein leichter benutzerdefinierter Fehler ist aufgetreten.	Das Roboterprogramm hat einen leichten Fehler gemeldet.	
C9200 : C9299	Ausgabe einer benutzerdefinierten Warnung	Das Roboterprogramm hat eine Warnmeldung ausgegeben.	

Tab. A-1: *Fehlercodes (29)*

Index

A

Accel-Befehl 6-8
Act-Befehl 6-10
Anweisung 5-1
  angehängte 4-45
  Aufbau 4-2
Ausgangsbitmuster
  festlegen 9-53
Ausgangssignale 4-38
Automatikbetrieb 3-44
Automatische Rückkehr 9-46

B

Base-Befehl 6-12
Basiskoordinatensystem 9-37
Batterie
  anzeigen 3-87
  Batteriezähler zurücksetzen 3-86
Befehle
  detaillierte Beschreibung 6-8
  Übersicht 6-2
Benutzerdefinierter Bereich 9-39
Beschleunigungszeit 4-13
Betriebsrechte 2-7
Bremszeit 4-13

C

CallP-Befehl 6-16
ChrSrch-Befehl 6-18
Close-Befehl 6-19
Clr-Befehl 6-20
CmpG-Befehl 6-30
Cmp-Jnt-Befehl 6-21
Cmp-Off-Befehl 6-29
Cmp-Pos-Befehl 6-23
Cmp-Tool-Befehl 6-26
Cnt-Befehl 6-32
ColChk-Befehl 6-36
ColLvl-Befehl 6-40
Com-Off-Befehl 6-42
Com-On-Befehl 6-43
Com-Stop-Befehl 6-44

D

Datentypen 5-5
Datum einstellen 3-88
Def-Act-Befehl 6-45
Def-Arch-Befehl 6-48
Def-Char-Befehl 6-50
Def-Double-Befehl 6-53
Def-Float-Befehl 6-53
Def-Fn-Befehl 6-51
Def-Inte-Befehl 6-53
Def-IO-Befehl 6-55
Def-Jnt-Befehl 6-57
Def-Long-Befehl 6-53
Def-Plt-Befehl 6-58
Def-Pos-Befehl 6-61
Dim-Befehl 6-62
Dly-Befehl 6-64

E

E/A-Link-Funktion 10-2
Echtzeitsteuerung 11-1
Ein-/Ausgänge
  externe 10-1
  spezielle 10-11
Einführung 1-1
Eingangssignale 4-37
Einschaltzeit anzeigen 3-87
End-Befehl 6-66
Error-Befehl 6-67
Externe Signale
  ein- und ausgeben 4-37
  Zeitablaufdiagramme 10-25
Externe Steuerung 11-1

F

Fehler
  anzeigen 3-78
  Codes A-2
  Diagnose A-1
  Übersicht A-2
  zurücksetzen 3-52
Feinpositionierung 4-16
Fine-Befehl 6-68
Fine-J-Befehl 6-70
Fine-P-Befehl 6-72
For-Next-Befehl 6-74

FPrm-Befehl 6-76
Funktionen
allgemeine Hinweise 8-1
detaillierte Beschreibung 8-2
Steuergerät 2-1
Teaching Box 2-5

G

Gelenkbremsen lösen 3-81
Gelenk-Interpolation 4-5
Geschwindigkeit 4-13
GoSub-Befehl 6-79
GOT1000
Anschluss 10-44
Technische Daten 10-45
GoTo-Befehl 6-80

H

Handgreifer
ausrichten 3-22
öffnen/schließen 3-21
steuern 4-20
Handgreiferzustand
Zustand nach Initialisierung 9-51
HClose-Befehl 6-82
Highspeed-RAM-Modus 9-71
Hlt-Befehl 6-81
HOpen-Befehl 6-82

I

If ... Then ... Else-Befehl 6-83
Input-Befehl 6-85
Interrupt 4-33

J

JOG-Betrieb 3-9
JOG-Betriebarten 3-9
JOvrd-Befehl 6-86
JRC-Befehl 6-87

K

Kollisionsüberwachung
Arbeitsweise 9-98
Befehle 6-36
Funktionsbeschreibung 9-98
Kommunikation
Einstellungen Ethernet 9-57
Einstellungen RS232 9-56
Funktionen 4-39
Konstanten
Einteilung 5-6
Gelenkkonstanten 5-10
Positionskonstanten 5-7
Zeichenkettenkonstanten 5-7
Kontinuierliche Bewegung 4-11
Kontrast einstellen 3-94
Koordinatensysteme 5-31
Kreis-Interpolation 4-9

L

Label-Befehl 6-90
Landessprache auswählen 3-91
Linear-Interpolation 4-7
LoadSet-Befehl 6-91
Logische Werte
Beschreibung 5-23

M

Marken 5-2
MELFA-BASIC V
Ausdrücke 4-41
Befehle 6-1
Begriffserklärung 5-1
Funktionsübersicht 4-1
Operationen 4-41
Menübaum 3-1
Menüpunkt auswählen 3-7
Monitor-Funktion
Ausgangsregister 3-74
Ausgangssignale 3-70
Eingangsregister 3-72
Eingangssignale 3-68
Variable 3-76
Mov-Befehl 6-93
Multirotationdaten
allgemeine Beschreibung 5-9

Multitasking-Funktion
Anwendung 4-54
Anwendungsbeispiel 4-55
Ausführung 4-48
Beschreibung 4-47
Programm erstellen 4-52
Programmplatzparameter 4-50
Mva-Befehl 6-95
Mvc-Befehl 6-97
Mvr2-Befehl 6-101
Mvr3-Befehl 6-103
Mvr-Befehl 6-99
Mvs-Befehl 6-105
MvTune-Befehl 6-108
Mxt-Befehl 6-111

N

NOT-HALT-Schalter
Anschluss 10-41

O

On-Com-GoSub-Befehl 6-115
On-GoSub-Befehl 6-117
On-GoTo-Befehl 6-119
Open-Befehl 6-121

P

Palettierungs-Funktion 4-22
Parameter
anwendungsspezifische Einteilung 9-1
auf Werkseinstellung zurücksetzen 3-85
einstellen 3-79
Plt-Befehl 6-125
Position
anfahren 3-35
Positionsdaten
ändern 3-36
eingeben 3-30
löschen 3-34
Prec-Befehl 6-128
Print-Befehl 6-129
Priority-Befehl 6-131
Programm
Aufbau 4-2
editieren 3-28
erstellen 3-25
kopieren 3-54
löschen 3-56
Namen ändern 3-55
schützen 3-58
Speicherkapazität 4-58
Programmplatzparameter
Beschreibung 4-50
Programmschleife 4-32
Programmschritt 5-2
Programmsteuerung
externe Signale 10-35
Verzweigungen 4-30

R

RAM-Modus 9-71
RelM-Befehl 6-132
Rem-Befehl 6-133
Reservierte Wörter 5-30
Reset-Err-Befehl 6-134
Return-Befehl 6-135
Roboter bewegen 3-9
Roboterbewegung
steuern 4-5
Roboterstatusvariablen
allgemeine Hinweise 7-1
detaillierte Beschreibung 7-2
Übersicht 5-18
ROM-Modus 9-71

S

Schnittstelle
serielle 9-56
Schrittnummern 5-2
Select-Case-Befehl 6-137
Servo-Befehl 6-139
Servospannung ein-/ausschalten 3-50
Sicherheitshinweise
grundlegende 6-I
Singulärer Punkt
Beschreibung 6-107
Durchfahren 9-92
Fehlermeldung 9-70
Skip-Befehl 6-140
Software-Version anzeigen 3-89
Spd-Befehl 6-141
SpdOpt-Befehl 6-143
SPS
E/A-Link-Funktion 10-2
Startmenü
Einstellungen 3-91
Stellungsmerker
beim Knickarmroboter 11-23
beim SCARA-Roboter 11-25
Beschreibung 5-8
Steuergerät
Bedien- und Signalelemente 2-1
Ethernet-Anschluss 10-46
Stopp 4-36

T

Teaching Box
- Bedienelemente 2-5
- Bedienung 3-1
- Display-Anzeigen 2-3
- Funktionen 2-3
- Menübaum 3-1

Testbetrieb 3-67
Timer 4-35
Title-Befehl 6-146
Tool-Befehl 6-147
Torq-Befehl 6-149

U

Uhrzeit einstellen 3-88

V

Variablen
- Ein/-Ausgangsvariablen 5-14
- Einteilung 5-11
- externe 5-15
- Feldvariablen 5-14
- Gelenkvariablen 5-13
- numerische 5-12
- Positionsvariablen 5-13
- Roboterstatusvariablen 5-18
- Verfahrweggenauigkeit 4-18
- Zeichenkettenvariablen 5-13

Verbindung
- überprüfen 9-66

Verfahrwegsgrenzen definieren 9-45
Verzweigungen 4-30

W

Wait-Befehl 6-151
Warmlaufbetrieb 9-83
Weltkoordinatensystem 9-37
While-Wend-Befehl 6-152
Wth-Befehl 6-153
WthIf-Befehl 6-154

X

XClr-Befehl 6-155
XRst-Befehl 6-157
XRun-Befehl 6-158

Z

Zeichen
- eingeben 3-5
- löschen 3-6
- mit besonderer Bedeutung 5-4
- Typen 5-3

Zeitablaufdiagramm 10-25
Zusatzfunktionen 3-90

DEUTSCHLAND

MITSUBISHI ELECTRIC
EUROPE B.V.
Gothaer Straße 8
D-40880 Ratingen
Telefon: (0 21 02) 4 86-0
Telefax: (0 21 02) 4 86-11 20
www.mitsubishi-automation.de

KUNDEN-TECHNOLOGIE-CENTER

MITSUBISHI ELECTRIC
EUROPE B.V.
Revierstraße 21
D-44379 Dortmund
Telefon: (02 31) 96 70 41-0
Telefax: (02 31) 96 70 41-41

MITSUBISHI ELECTRIC
EUROPE B.V.
Kurze Straße 40
D-70794 Filderstadt
Telefon: (07 11) 77 05 98-0
Telefax: (07 11) 77 05 98-79

MITSUBISHI ELECTRIC
EUROPE B.V.
Lilienthalstraße 2 a
D-85399 Hallbergmoos
Telefon: (08 11) 99 87 4-0
Telefax: (08 11) 99 87 4-10

ÖSTERREICH

GEVA
Wiener Straße 89
AT-2500 Baden
Telefon: (0 22 52) 8 55 52-0
Telefax: (0 22 52) 4 88 60

SCHWEIZ

Robotronic AG
Schlachthofstrasse 8
CH-8406 Winterthur
Telefon: (0 52) 267 02 00
Telefax: (0 52) 267 02 01